Illuminance Calculation Algorithm and Hardware for Simultaneous Sensing and Energy Harvesting Using Solar Cell Harvester at Maximum Power Point

By

Olmar van Beurden
4368150
O.vanBeurden@student.tudelft.nl

to obtain the degree of Master of Science
in Electrical Engineering
Microelectronics
at the Delft University of Technology
to be defended publicly on 27-08-2021 at 10:00.

Thesis committee:

Prof.dr.ir. W.A. Serdijn	TU Delft, supervisor
Dr.ir. T.G.R.M. van Leuken	TU Delft
Dr. M. Dezyani	NOWI, daily supervisor

Delft University of Technology
Faculty of Electrical Engineering, Mathematics, and Computer Science
Mekelweg 4
Delft, The Netherlands

Contents

Abstract

Internet-of-things (IoT) devices utilising energy harvesting often use solar cells as their harvester. The same solar cells can also be used to sense illuminance (in lux). Current and voltage sensing systems are researched. Also, the accuracy limits of the system are researched to see what accuracy is expected. Different design options, based on the maximum power point (MPP), open-circuit voltage, and short-circuit current are compared. A design based on floating-point (FP) logic and a look-up table (LUT) are made in Verilog. The final LUT design is synthesised using Cadence Genus and verified using an FPGA and Cadence Spectre circuit simulator. The design has an area of 1.46 mm^2 and has an average power consumption of 121.68 μW. It can sense illuminance with an average error of 7.0% and a maximum error of 15.9%, in a range of 188 - 6130 lx for an indoor solar cell, and with an average error of 4.7% and a maximum error of 13.5% in a range of 272 - 43400 lx for an outdoor solar cell.

Glossary

α	Short-circuit current temperature coefficient (A/°C)
A	Solar cell diode ideality factor
AC	Alternating current
ADC	Analog-to-digital converter
β	Open-circuit voltage temperature coefficient (V/°C)
B	Charge pump conversion ratio
C_{input}	Current sensor input capacitance
C_{ox}	MOSFET oxide capacitance per unit area
D	Ratio between I_{mpp} and I_{sc}
DAC	Digital-to-analog converter
DC	Direct current
ENOB	Effective number of bits
FoM	Figure of merit
FP	Floating-point
FSM	Finite-state machine
f_{sw}	Switched-capacitor power converter switching frequency
G	Illuminance (in lux)
G_0	Illuminance (in lux) under standard test conditions
I_{avg}	Switched-capacitor power converter average current
I_{mpp}	Solar cell output maximum power point current
I_o	Dark saturation current
I_{ph}	Photo generated current (see Figure 3.2)
I_{rip}	Switched-capacitor power converter output ripple current
I_{sc}	Solar cell output short-circuit current
IC	Integrated circuit
IoT	Internet-of-things
k	Boltzmann constant $1.380649 \times 10^{-23} \frac{\text{J}}{\text{K}}$
L	MOSFET channel length
LSEB	Least significant effective bit
LUT	Look-up table
λ	Channel-length modulation parameter
M	SenseFET current division ratio
MOCD	MOSFET-only current divider
MPP	Maximum power point
MPPT	Maximum power point tracker
μ	Charge-carrier effective mobility
n_s	Number of solar cells in series
PMIC	Power management integrated circuit
PVT	Process, voltage and temperature
q	Elementary charge constant, $1.602176634 \times 10^{-19}$C
R_{ds}	MOSFET drain-source resistance
R_g	Current sense resistor
R_s	Solar cell series resistance (see Figure 3.2)
R_{sh}	Solar cell shunt resistance (see Figure 3.2)
SAR	Successive-approximation-register (analog-to-digital converter)

SCPC	Switched-capacitor power converter
STC	Standard test conditions
T	Temperature (in °C)
V_{ADC}	Voltage as measured by the analog-to-digital converter
V_{ds}	MOSFET drain-source voltage
V_{mpp}	Solar cell output voltage at maximum power point
V_{oc}	Solar cell output open-circuit voltage
V_{off}	SenseFET amplifier offset voltage
V_{sd}	MOSFET source-drain voltage
V_{sg}	MOSFET source-gate voltage
V_T	MOSFET threshold voltage
V_t	Thermal voltage
W	MOSFET channel width

1. Introduction

1.1 Project motivation

The internet-of-things (IoT) gets bigger, and each device needs energy. This energy can be delivered by batteries, however, these need replacement, or need to be recharged somehow. To keep devices working remotely without the need for maintenance, energy harvesting techniques can be used to keep the batteries charged.
Energy harvesting is energy critical, as only a little power can sometimes be harvested from the environment. Furthermore, to keep costs down, as few external components as possible should be included.
Integrating more functions in the same hardware can be profitable for both of these requirements. Often one of the functions of a remote IoT device is sensing parameters from its environment, in which case integrating a sensing system within the energy harvester is beneficial, as fewer external components are needed.

The question is whether it is possible to relate the incident power of a maximum power point tracker (MPPT) to a physical quantity, like irradiance, illuminance, temperature, or vibration, instead of using a separate sensor for this.

The type of information that can be gathered depends on the transducer in the harvester, like a solar cell, a piezo element, or a thermoelectric generator.
Different physical quantities have very different properties. This project focuses on illuminance via a solar cell, since solar cells are one of the most widely used harvester types, and it is what the NOWI NH16, the chip in which this research should be integrated, is based on.

1.2 Irradiance or illuminance

Irradiance is a measure of light power and is given in W/m^2. Irradiance is the intensity with which the light source shines.

Illuminance is luminous flux, or how much the incident light illuminates a surface, which is different from irradiance, as illuminance is corrected for how light is perceived by the human eye. In other words, illuminance is the brightness of the light as perceived by a human. So the difference with irradiance is that only visible light adds to illuminance, and infra-red and ultraviolet light do not. The unit of illuminance is lux.

Depending on the application either irradiance or illuminance is required. In this project, the focus will be on illuminance.

1.3 Research question

This leads to the following research question:

Can a low-power energy harvester MPPT be used as a sensor front end for sensing illuminance?

In other words, can the measurements that are done within the MPPT be related to the illuminance of the solar cell?

1.4 Report overview

The goal of the thesis is to show how the NOWI NH16 power management IC (PMIC) can be used as a sensor front-end.

The main function of the NOWI NH16 is to function as an MPPT, however, with minimal power usage and footprint increase, the device should be able to estimate the illuminance. To do so, this thesis looks as follows.

Chapter 2 gives an overview of the state of the art of self-powered illuminance sensors, other self-powered sensors, and the NOWI NH16 chip, in which this system needs to be integrated.

Also, it gives a literature study of current sensing techniques, and voltage division techniques. The effect of each part of the system on the resulting accuracy is discussed, and a breakdown of the most critical parts for accuracy is given.

Chapter 3 discusses the higher-level system design of the system designed in this project. We give several options to calculate the illuminance using measured current, voltage, or both. Subsequently, we make a trade-off between these methods to decide the best method, comparing accuracy, computational complexity, energy efficiency, and possible integration with the NOWI NH16. We implement the best option in the next chapter.

Chapter 4 discusses the way to implement the chosen model in hardware. We discuss a floating-point design and a logarithmic design. We further investigate the logarithmic design and we design and compare two different implementations. After gate-level synthesis, we choose the best implementation based on area and power.

Chapter 5 discusses the verification of the system. We verify the system both via event-based simulation, and circuit simulation. We do circuit simulation with the designed system on its own, and also with the NOWI charge pump, current sensor and analog-to-digital converter (ADC) added. Lastly, we verify the system on an FPGA. We use both a solar cell model and measured solar cell data as inputs.

Chapter 6 gives a conclusion, summarising the results of the project. The main contributions of this project are listed and recommendations for future work are discussed.

2. Literature

2.1 Prior art

In this section, we investigate previous work. Included are sensors measuring illuminance, and sensors that harvest energy via the same transducers as they use for sensing.

2.1.1 Illuminance sensors

The system will sense illuminance. It is therefore interesting to compare the system to illuminance sensors, like ambient light sensors. Ambient light sensors measure short-circuit current which they then link to the illuminance.

An example is the MAX44007 by Maxim Integrated [1]. This sensor has a dynamic range of 0.025lx to $104,448$lx, with less than $1\ \mu$A current consumption, and 22 bits. It mentions a total error of 15% maximum.
Another ambient light sensor is the Texas Instruments OPT3001 [2]. This one has a smaller dynamic range of 0.01 to 83 klx. For lower illuminance conditions it mentions a linearity error of about 5%.

More advanced devices to measure illuminance are lux meters and spectrometers. These are the most accurate devices to measure illuminance with. They have an inaccuracy range of about 2-7% [3].

In [4], the use of illuminance sensors for smart lighting is investigated. A summary of different types of low-cost illuminance sensors is given, mentioning digital sensors, with everything integrated, analog sensors, which still require an ADC, and photodiodes, which also require an ADC. The error of these low-cost sensors according to [4] is more than 15%, which is not good enough for smart lighting. This paper then suggests the use of a better sensor, including a 24-bit resolution ADC and noise filter. With this, in [4], an inaccuracy of 4% for outdoor use and 1% for indoor use is reached.

Though the accuracy given in [4] is good, it requires a very high-resolution ADC, which is expensive, and power hungry. The system therefore cannot be used for energy harvesting.

2.1.2 Pyranometers

Pyranometers are made to measure light energy in W/m^2. Multiple types exist:

- Solar cell pyranometers
- Blackbody thermopile pyranometers

Solar cell pyranometers are less accurate, mostly in cloudy conditions, but cheaper than blackbody thermopile pyranometers. Good pyranometers have a resolution down to $1\ \mathrm{W}/\mathrm{m}^2$, and an inaccuracy down to 3%, however, on a cloudy day, a solar cell pyranometer can have errors of about 8% compared to blackbody thermopile pyranometers. Blackbody pyranometers would not be able to harvest energy, as they convert the incident energy into heat to do the measurement.

2.1.3 Self-powered sensors, two transducers

Many self-powered sensors, including light sensors, exist, however, most combine an energy harvester with a separate sensor front end.

In [5], an active pixel sensor with a photodiode integrated on the same chip area is described. The photodiode is used to power the chip with active pixel sensors. Interestingly, both are integrated into the same chip, however, this paper still uses two different devices for sensing and harvesting. Also, the integrated photodiodes can only constitute a small area, and thus this will likely not be sufficient for primarily harvesting energy.

In [6], a self-powered illuminance sensor is described. However, it does use a solar cell and a light sensor for energy harvesting and illuminance sensing respectively. The accuracy is not given in [6], but it uses an Avago APDS-9004 analog output illuminance sensor [7].

Many more examples exist of sensing systems that contain a solar cell for energy harvesting, like [8] and [9], however, the point of this project is to use only one transducer for both harvesting and sensing, and thus these papers are not the most interesting.

2.1.4 Self-powered sensors, single transducer

Some sensor products using photodiodes for sensing also use the same photodiodes for harvesting energy. The harvested energy is then used to power the sensor.

Ring oscillator based self-powered illuminance sensor

In [10], a solar cell based self-powered illuminance sensor is described. The sensor works based on its voltage. Since the solar cell is supplying power to the system, it does not work at open-circuit voltage exactly. A ring oscillator is powered via the solar cell, which starts oscillating with its frequency dependent on the voltage. The output graph given in [10] shows a more or less linear line on a log-log plot between the oscillator frequency and illuminance. No accuracy is given, however.

To get some idea of the accuracy of the sensor, we extract the data from the figure given in the paper, make a linear fit, and then calculate the error. The error has been plotted in Figure 2.1.

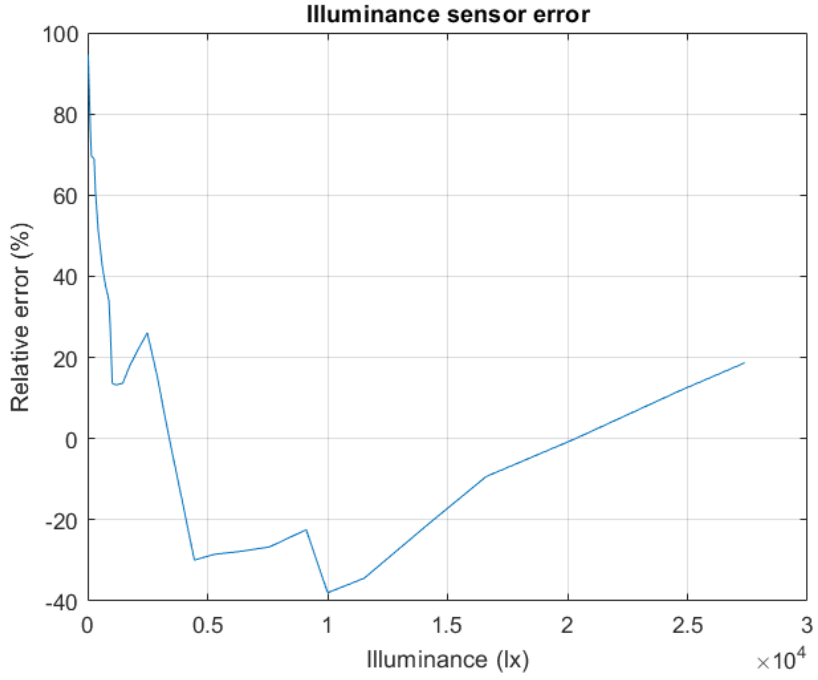

Figure 2.1: Calculated error from the illuminance sensor of [10]

Disregarding the first part, the sensor has an error of up to 40%. The average error can be calculated and is about 20%.

The sensor is self-powered, but no power is given. Used in the design is a TOSHIBA TC7WU04FK ring oscillator, which consumes about $6\ \mu$W. This makes our best power consumption estimate to be at least $6\ \mu$W.

Battery-charging-chip based self-powered illuminance sensor

Another example where the harvester is used as the sensor front-end can be found in [11], in which Texas Instruments has designed a self-powered ambient light sensor using a solar cell, a battery charging chip, and a LED used as load. In [11], the solar cell charges a capacitor, which is then discharged by the LED. The higher the input power, and thus illuminance, the faster the charging and thus the cycle time. Based on the frequency of this pulse an illuminance estimation is made.

Integration with an MPPT would be difficult as the MPPT changes the current, resulting in a different output frequency. Lastly, it is hard to determine the accuracy of this system as no clear figure has been given in the report.

Self-powered sun sensor

In [12], a self-powered sun sensor is described, which uses its photosensors as energy harvesting transducers. This paper makes an improvement over standard sun sensors by using a miniature sundial. A sundial has a larger illuminated area compared to standard sun sensors, and this increases harvested energy. The estimated average inaccuracy is about 5%, with a maximum error of up to 11%, but it has only a few measured points. In [12], reasonable accuracy is reached, but the measured quantity, sun angle, is different from the focus of this project. Also, the focus is solely on powering the sensor and not on having energy left to harvest, though it probably could.

Self-powered light direction sensor

Similar to the previous paper, in [13], a self-powered light direction sensor is described. The sensor consists of photodiodes and a wall blocking part of the incoming light. Depending on the photocurrents in the photodiodes, the angle of the light is determined. The light angle can be detected with an accuracy of 7 ENOB, which would correspond to an error of less than 1%. In [13], it is mentioned that the sensor requires a light intensity of $25\text{mW}/\text{cm}^2$ to operate. Compared to this project, in [13] light angle is measured instead of illuminance.

Self-powered gas sensor

In [14], a gas sensor is described that used a solar cell and a colourimetric film. In the presence of NO_2, the colourimetric film reacts, changing the amount of light reaching the solar cell. The change in current is measured, and based on this change the concentration of NO_2 is estimated. The sensor is self-powered but is not used to harvest energy. Also, the quantity measured in [14] is different from what we need for this project.

Self-powered image sensor

In [15], a CMOS image sensor is described that gets part of the energy necessary by first charging pixels via the photodiodes. Also, if light levels vary over the pixels, if one is charged, it can then help charge surrounding pixels. Just like in [12], in [15], energy is only harvested to power the sensor and thus the system is not used to harvest energy. In fact, the sensor of [15] still requires external energy.

Self-powering home light detection

The Leviton WSCPC [16] is a series of self-powered light sensors to be used in combination with other Leviton products. This sensor uses solar cells both for light detection and to harvest enough energy to power its wireless module. The sensor is made to detect light on or off and thus does not give an exact value of light intensity.

A similar product is the Molex 180997-0006 [17]. This product is also meant to be used with an RF light switch, with the same drawback of not reporting an actual value of light intensity.

2.1.5 Energy harvesting sensors

Our system is supposed to be used for both sensing and harvesting at the same time. Therefore it is interesting to compare our system to current systems that do both.

Mobile device ambient light sensor harvesting

In patent [18], the idea is to use small solar cells as ambient light sensors for mobile devices. The solar cells are placed behind the screen and in reaction to illuminance, the brightness is adjusted. The patent also mentions that when the solar cells are not used for sensing they can be used to charge a battery.

Accuracy is not mentioned, and also the solar cells are not in an optimal position to harvest energy, as they are behind several layers of film.

Active pixel sensor energy harvesting

In [19], harvesting energy with active pixel sensors during idle time is mentioned. A CMOS pixel array is used as the transducer and includes a boost converter with an MPPT algorithm, to charge the battery. The main difference between [19] and this project, is that the main purpose of [19] is image sensing, with energy harvesting as a bonus.

In [20], a pixel array with photodiodes is created, of which the photodiodes are reconfigurable for either energy harvesting or image sensing. When part of the array is used for energy harvesting, the other part can be used for image sensing. Cubic spline interpolation can fill in the gaps left by using part of the array for energy harvesting.

Energy harvesters as sensors

In [21], the data of different energy harvesters is used for place recognition. In this work solar cells, piezo elements, and Peltier elements are used. The harvesters are only used for sensing in this case, however, in future work, using them for both harvesting and sensing is suggested.

2.1.6 NOWI NH16

The design is supposed to be integrated into the current NOWI system. The NOWI system is an energy harvesting chip or, in other words, a small footprint, ultra-low power DC/DC converter with MPPT functionality. The NOWI energy harvesting chip consists of the following components:

- Charge pump (DC/DC converter)
- Input capacitor for low pass filtering
- Current divider
- Current mirror
- Resistor for current to voltage conversion
- ADC
- FSM for MPPT control
- Several other digital control circuits like counters and oscillators
- Battery

A diagram of the system can be found in Figure 2.2.

Charge pump

The charge pump is the actual DC/DC converter. By changing the way the capacitors in the charge pump are switched, the conversion ratio of the system is altered. In this case, 16 different conversion ratios exist.

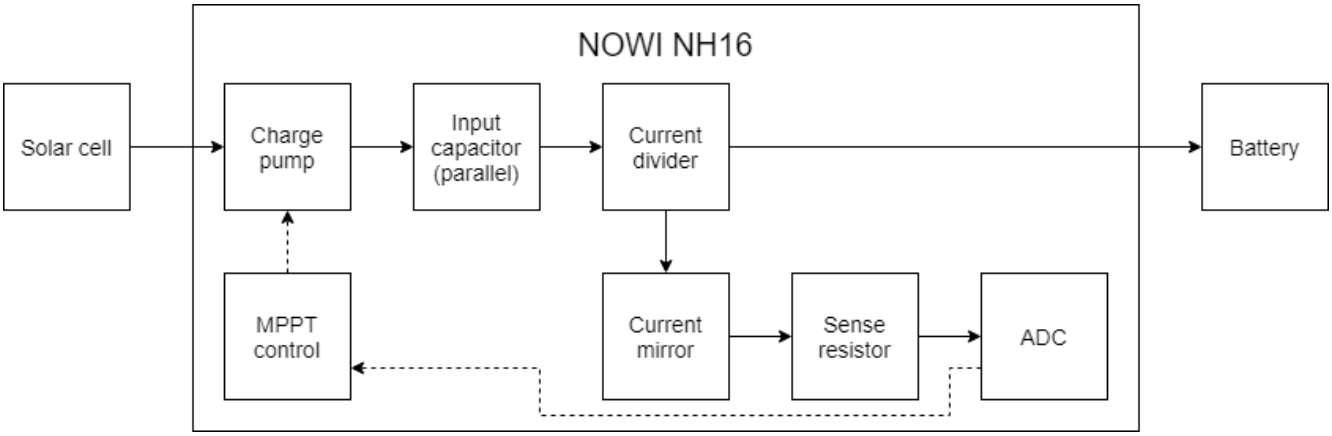

Figure 2.2: NOWI NH16 block diagram

Input capacitor

The input capacitor together with resistance R_s, the series resistance of the senseFET, form a low-pass filter. The low-pass filter is supposed to filter out ripples generated by the charge pump due to switching. The bigger this capacitance the better it filters the ripple, according to Equation 2.1.

$$I_{rip} = \frac{I_{avg}}{f_{sw} R_s C_{input}} \tag{2.1}$$

In Equation 2.1 I_{rip} is the charge pump output ripple current, I_{avg} is the charge pump average output current, f_{sw} is the charge pump switching frequency, R_s is the on-resistance of the senseFET, and C_{input} is the input capacitance.

But a larger capacitance also adds delay, which in this case is mainly determined by the time constant of this capacitor together with R_s, or $\tau = R_s \times C_{MPPT}$.

R_s varies with supply voltage. To get the best performance considering ripple voltage and delay, there are two possible input capacitors, a small one, and a large one that can be connected separately.

Current divider

Current division is done via a senseFET structure, also known as a MOSFET-only current divider (MOCD). It is a current division circuit, consisting of two MOSFETS of which one is M times as wide as the other. An amplifier makes sure the drain-source voltage of the MOSFETs is the same, and thus their drain currents are proportional to each other with factor M. The senseFET is further explained in Section 2.2.3.

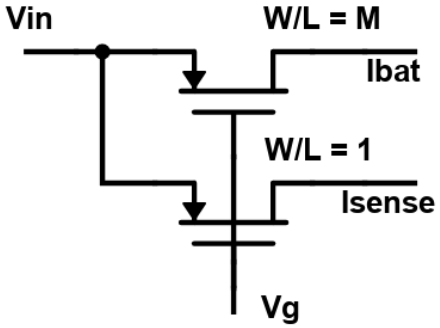

Figure 2.3: SenseFET circuit

Current mirror

A current mirror copies the sense current from the senseFET structure and sends it through a sense resistor R_g which creates a voltage, which is measured by the ADC.

The senseFET ratio and voltage generating resistor R_g can be adjusted such that the voltage over $R_g < 0.8\,\text{V}$, which is the reference voltage for the ADC, and thus its the maximum allowed input voltage.

Sense resistor R_g

The sense resistor is variable in resistance depending on the current range and is implemented by multiple series resistances that can be bypassed via switches.

MPPT algorithm

A lot of MPPT algorithms exist. The algorithm used in the NOWI chip is perturb and observe (P&O). The system controls both the input/output voltage conversion ratio and the switching frequency.

For deciding if a perturbation has resulted in a higher power, the system measures the output current and compares it to the current of the previous sample. Since there is a battery at the output the output voltage can be assumed to be constant between two measurements, and thus comparing currents is enough to determine if power increased or decreased.

Battery

The system has a battery to store energy. This is necessary to keep the output voltage constant. A capacitor in theory could also be used but will have bigger voltage swings.

2.1.7 Conclusion

In Chapter 2.1, we investigated previous art and the current state of the art. From this, we conclude that similar work exists, but not exactly as we do in this project. Self-harvesting sensors exist, even ones that sense illuminance, however, none of them have energy harvesting as the primary function. Most of them only use the harvested energy directly for sensing, but nothing else. Closest to sensors that both sense and harvest are pixel arrays, but these have a different function than sensing illuminance. This project, therefore, enters novel territory by using an energy harvester for simultaneously sensing illuminance, as opposed to harvesting energy with something that is used primarily as a sensor.

Furthermore, we explained the NOWI NH16, in which our research will be integrated. Next up is a breakdown of different sensing techniques that can be used in combination with the NOWI NH16 to turn the system into an illuminance sensor. Also, we give the effect of each part of the NOWI NH16 on the overall system's accuracy.

2.2 Current sensing

The first step to calculating the illuminance is sensing an electrical quantity. This could be voltage or current, but can also be magnetic flux or charge. In the end, we want to visualise the illuminance on a screen. To do this, the output of the system is a binary value, represented via bits, which are in turn represented by a voltage. However, getting to that voltage can be done via several steps, through different electrical quantities, or even other energy domains. In this section, we discuss different ways to sense current. We then use the sensed current to estimate the illuminance.

Different methods of current sensing are described in [22], and another one is described in [23].
In [24], current sensors are categorised according to one of four principles:

- Ohm's law (resistance)
- Faraday's law of induction

- Magnetic field sensors
- Faraday effect

In this project, we classify current sensors according to what electrical domains are used for sensing. For electrical sensing, four main domains exist:

- Voltage
- Current
- Charge
- Magnetic flux

This method of classification allows for a wider range of sensors than described in [24].

Different current sensing techniques are:

- Direct current to voltage translation: resistor based, or inductor voltage integration based.
- Making a copy of (part of) the current and measuring this: current to current conversion. This then needs to be translated to voltage, which can be done in the ways described above.
- Charge based: capacitor voltage differentiation.
- Magnetic-flux based: magnetic field to voltage conversion, via magnetic field sensors or the Faraday effect.

A summary of possible current sensing techniques based on this classification can be found in Table 2.1.

Table 2.1: Current sensing method classification table

Direct current to voltage or current to digital	**Magnetic field**	**Part of the current**	**Charge**
Sense resistor [25]	Hall effect [24], [26]	SenseFET [27], [28], [29], [30], [31]	Capacitor voltage differentiation [23], [32]
R_{ds} based [33], [34]	Fluxgate principle [24]	Current transformer [24]	
Inductor DC resistance [28], [33], [35]	Magnetoresistor [36]		
Inductor voltage integration (observer-based approach) [37], [38]	Piezo cantilever [39], [40]		
Current DAC SAR [41]	Rogowski coil [24], [42]		
	Faraday effect [24]		

2.2.1 Sense resistor based

This method uses a sense resistor to transform current into voltage, according to $U = I \times R$. A sense resistor is a resistor with a very precisely known value, and with a small voltage drop. The voltage is measured, and from there the current can be calculated. This method can be very accurate, however will also be quite power hungry, since you add a resistor in the current path. There is a direct trade-off between accuracy and power consumption by increasing or decreasing the resistance.

At higher frequencies, resistors can show inductive behaviour, which would increase the resistance and thus reduce the accuracy. However, the NOWI NH16 has a relatively low frequency and thus this phenomenon should not have an effect.

The inaccuracy of a sensing resistor in an integrated circuit (IC) can be as low as 0.1% at room temperature and 0.15% over a wide temperature range if temperature correction is applied to counteract the resistor's temperature coefficient [25].

2.2.2 R_{ds} based

This method essentially works the same as the previous method, but uses the resistance of a switch that is already in the system, and thus would not cost any additional power. In the case of a low drain-source voltage, a MOSFETs resistance can be calculated according to Equation 2.2.

$$R_{ds} = \frac{L}{W\mu C_{ox}(V_{gs} - V_T)} \tag{2.2}$$

Equation 2.2 is an approximation. In reality, R_{ds} is non-linear and depends heavily on PVT variations [33]. It is therefore not a very well defined resistance. Calibrating the system can increase accuracy, however, this method will not be as accurate as using a sense resistor. In theory, not only R_{ds} but also any other form of resistance in the current path can be used, for example, traces, as long as their resistance value is well known, and if they are big enough that they produce a measurable voltage within the current range. Effects like thermal drift and process variations limit accuracy.
In [34], a calibration method is proposed, using a precision sense resistor to calibrate R_{ds}, making the system almost as accurate as the sense-resistor method, at the cost of more required chip space and complexity. They report an inaccuracy of 22.6% without calibration, up to 0.45% with calibration.
In [33], an inaccuracy of about 10% is mentioned.

2.2.3 SenseFET based

This method again uses a resistor to transform current into a voltage, but first copies a fraction of the current to another branch. Decreasing the current that passes through the resistor reduces the power consumption, however, it also introduces another inaccuracy into the system, depending on the matching of MOSFETs. The circuit of the senseFET structure can be seen in Figure 2.3.

This option gives some extra ways to tune the system since by changing the current ratio the power consumption and accuracy can be traded off. In theory, this method can be combined with any of the current sensing methods, but only makes sense to combine with accurate but power inefficient methods. This is the system that will be used in the NOWI NWA16x.

SenseFET inaccuracy can be caused by drift due to temperature, parasitics causing inaccuracy in the MOSFETs ratio, or if the sense resistor's value is non-negligible compared to the senseFET [27]. Mismatch can be compensated by using larger devices, but this also increases power consumption, thus resulting in another trade-off between accuracy and power consumption. Another source of inaccuracy can come from amplifier offset [28]. During the startup of MOSFETs, transients can be measured which can also have an impact on the senseFET accuracy, however, this mostly seems to be an issue in power MOSFETs at higher voltages [29].

The senseFET method can be improved by using error averaging techniques [28]. This technique can be used to average errors over multiple MOSFETs, and thus reduce the final error. This technique requires little extra power, but does require more chip area, and can only work if enough bandwidth is available. They mention an inaccuracy of 2.8%.

Naturally, since this method still depends on a sensing resistor it cannot be more accurate than the sense resistor method. In [30], an inaccuracy down to 1% is reported. Instead of using a sensing resistor, the senseFET method can also be combined with the R_{ds} method. In [31], temperature coupling between the two MOSFETs is used to improve the senseFET accuracy over a bigger temperature range. An inaccuracy of 8% is reported.

The accuracy of this method depends on the MOSFETs aspect ratio, but can never be better than the sensing method used at the end, like a sense resistor or the R_{ds} method. The biggest advantage of this method is that it can save power.

2.2.4 Current-transformer based

The idea of this sensing method is to copy (a part of) the current to another branch in which you measure this current. The main advantage is that this method is isolated and has low thermal drift. Also, since you can divide the current, you can choose the amount of current running through the secondary coil and have a direct trade-off between power and accuracy.
Disadvantages are that another method to actually measure the current, like a sensing resistor, must be added to the secondary side. Also, transformers are relatively big, DC offset can saturate the coil, and it is not capable of measuring DC. The inaccuracy of this method is 0.1%-1% [24].

2.2.5 Inductor DC resistance

If an inductor is used in the DC/DC converter, then if you know the equivalent series resistance, this can be used to measure the current through the inductor. This method is also referred to as DC resistance current sensing. In this case, a low-pass filter is used to filter the voltage across the inductor and equivalent series resistance, and then by measuring the voltage over the low-pass filter capacitor the current can be calculated. This method has poor accuracy due to component tolerances and a large temperature coefficient. Also, to get sufficient accuracy, an inductor with enough DC resistance is needed, which means it is not entirely lossless, especially at higher temperatures [33].

Another accuracy limitation of this method is caused by leakage current to the sense pins. This can be corrected by balancing the voltage drop over the resistors in the sensing circuit [35].

In theory, this method is lossless if an inductor is already present in the system. However, the inductance and equivalent series resistance should be known accurately. In [33], an inaccuracy of about 5% is mentioned for this method due to component tolerances.

2.2.6 Inductor voltage integration based

The current through an inductor can be described by $v = L\frac{di}{dt}$, thus by integrating the voltage over the inductor, the current can be calculated [37]. This method is also called sensorless, or observer-based method.

This method has been reported to reach inaccuracies down to 9% [38].

2.2.7 Magnetic field sensors

Magnetic field sensors can also be used to measure current, with the added advantage that they can also measure static magnetic fields and thus measure DC currents. Magnetic field sensors can be based on:

- The Hall effect
- The fluxgate principle
- The magnetoresistance effect

Hall effect based

When current flows through a conductive material in a magnetic field, a voltage is generated perpendicular to the direction of current and the magnetic field according to $v = \frac{IB}{nqd}$, in which I is the current, B the magnetic flux density, n the charge carrier density, q the current carrier charge, and d the thickness of the material. A smaller thickness creates a larger voltage, however, also has more resistance and thus increases power loss [24]. Displacement relative to the current carrier, and offset can lower the accuracy of this method. Also, the method can suffer from high thermal drift.

Hall-effect based current sensors can have an inaccuracy down to 0.5% [26].

Fluxgate-principle based

The fluxgate principle is based on the non-linear relation between magnetic field H and magnetic flux density B within a magnetic material. An AC signal is used to create a magnetic field, which saturates the core. This magnetic field creates a voltage in a pickup winding. The first harmonic represents the excitation AC. The second harmonic can be extracted and its voltage is proportional to the external magnetic field, which in turn is proportional to the current you want to measure.

The main advantages are its very high accuracy, and that it is an isolated system. However, the system is rather large, needs an external sinusoidal current source, and requires complex control electronics.

The inaccuracy of this method can be as low as 0.001% [24].

Magneto-resistance based

The magneto-resistance effect is based on materials that change resistance under the influence of a magnetic field. This effect is divided into three categories:

- Anisotropic magnetoresistance (AMR)
- Giant magnetoresistance (GMR)
- Tunneling magnetoresistance (TMR)

The difference between the three is that GMR has a higher responsivity to magnetic fields than AMR and can thus detect lower fields. TMR is even more sensitive and thus can detect even lower fields [36]. The disadvantage of GMR and TMR is that they saturate at a lower magnetic field strength. All three suffer a lot from thermal drift, which can be somewhat overcome by a Wheatstone bridge, and non-linearity. Also, another medium is needed to actually sense the resistance, which means a well-known current source is needed.

In [24], an inaccuracy down to 0.5% is reported for this method.

2.2.8 Piezoelectric cantilever based sensing

This method, described in [39] and more in-depth in [40], uses a piezoelectric cantilever with a magnet attached to sense the magnetic field created by a current. The magnetic field bends the cantilever and the piezoelectric element will have a different charge depending on this displacement, resulting in a displacement-dependent voltage. The measured voltage is then correlated to the current. The main accuracy limitations are position and orientation errors, which are caused by the relative position and orientation of the sensor to the current-carrying wire.

The system is isolated, but is difficult to produce, and requires the use of a permanent magnet.

DC measurements are possible with a piezoelectric cantilever, however, the excursion is attenuated over time. This method is more precise for AC measurements since in that case it can be used to average the current over time. An inaccuracy of around 3% can be achieved with this method. Positioning schemes can improve inaccuracy to under 1% [43].

This sensing method is capable of measuring currents down to the mA range [44].

2.2.9 Rogowski coil based

Current sensing with a Rogowski coil is based on Faraday's law of induction. The idea is that any electrical current creates a magnetic field, which can then be measured. A coil picks up the magnetic field, and the induced current is then measured. This then is proportional to the original current. The main advantage of this method is that it is completely isolated from the system, which can be interesting for high-voltage applications. However, it can be difficult to get sufficient accuracy from smaller currents. Also, the principle is based on measuring flux change, which means this method is incapable of measuring DC currents.

The inaccuracy is about 0.2% up to 5% [24]. In [42], an inaccuracy of down to 0.8% is mentioned.

2.2.10 Faraday effect

The Faraday effect is based on the fact that applying a magnetic field can cause materials to induce circular birefringence. The polarisation of light waves through optic fibres is measured, in which a difference in the polarisation indicates the presence of a magnetic field, and thus of a current. The main accuracy limitation of this method is bending stress.

The main advantages of this method are low thermal drift, the system is isolated, and it can be very precise. However, due to its complexity, it is hard to measure small currents with this method. Methods exist to get it more precise for smaller currents but they involve expensive fibre optic cables to avoid bending stress.

The inaccuracy of this method is about 0.1%-1% [24].

2.2.11 Capacitor voltage differentiation based

A capacitor-based current sensing method has been described in [23]. The main idea is based on $I = C\frac{dv}{dt}$, in which I is the current through the capacitor, C is the capacitance, and $\frac{dv}{dt}$ is the change in voltage over the

capacitance over time. The capacitor is charged to a predefined voltage, and then discharged. The time this takes is measured, the capacitance is known beforehand, by which the current can be calculated. Inaccuracies in [23] are the capacitance and resistor component tolerances, the voltage-to-time converter, and the voltage references, or the comparator comparing them. The method is lossless and isolated. The same idea has been used to compare power for an MPPT and thus essentially created a power sensor [32]. This method has a reported inaccuracy of 7.6%, with an offset cancellation technique [23].

2.2.12 Current DAC based

Another way to sense current is by using an ADC controlled via a current DAC. Instead of comparing voltages within the successive-approximation-register (SAR) ADC, this system directly compares the current to reference currents. The advantage is that no current to voltage translating element is needed, saving power, while still having the potential to be very accurate. Instead, the system uses direct current-to-digital translation. This system does require a precise current reference.

The inaccuracy of this system can be as good as 0.8% [41].

2.2.13 Conclusion

A trade-off of the current sensing methods can be seen in Table 2.2. Looking at this table, several sensing options can be eliminated because their current range is not precise enough, some options do not integrate into silicon, are not DC capable, or require the use of an inductor. Good options that are left are by using a sense resistor, or via a senseFET. The R_{ds} method can be good, but only with calibration, which makes the system complex. Capacitor differentiation is not very accurate, and the current DAC SAR requires a precise current reference, which also makes the system complex.

The NOWI NH16 uses the senseFET method, which according to Table 2.2, is the best method for this project, as it saves power compared to using a sense resistor.

Table 2.2: Current sensing techniques comparison table

Method	Inaccuracy	Current range	Silicon integration	DC capable	Inductor required	Power consumption
Sense resistor	0.10%	mA	Y	Y	N	High
Rds	0.45% - 10%	mA	Y	Y	N	Low
SenseFET	1%	mA	Y	Y	N	Trade-off
Current transformer	0.1% - 1%	A	N	N	N	Trade-off
LDCR	5%	mA	Y	Y	Y	Low
Inductor voltage integration	9%	mA	Y	Y	Y	Low
Hall effect	0.5%	A	Y	Y	N	Low
Fluxgate	0.001% - 0.5%	mA	N	Y	N	Low
Magnetoresistor	0.50%	mA	Y	Y	N	Low
Piezo cantilever	3%	mA	N	Y	N	Low
Rogowski coil	0.2% - 5%	A	N	N	N	Low
Faraday effect	0.1% - 1%	kA	N	Y	N	Medium
Capacitor differentiation	7.60%	mA	Y	Y	N	Low
Current DAC SAR	0.80%	mA	Y	Y	N	Low

2.3 Power sensing

For a first approximation of illuminance, we can use power. A way to measure power directly is described in [45]. The sensor described measures the time of a voltage ripple on a capacitor by using a time-to-digital converter (TDC). It is comparable to the current measurement described in [23], but [45] also uses the average voltage to make it a power measurement. They mention a 14.1% gain error and 9.4% offset error, which is not accurate enough for our purpose.

2.4 Voltage division

The voltage that can be sensed is limited by the ADC reference voltage at $0.8\,\mathrm{V}$. If the voltage is too high, a voltage division is necessary before analog-to-digital conversion.

Possible voltage division circuits are [46]:

- Resistive divider
- Off-state MOS divider
- Diode stack divider
- Switched capacitor voltage divider

2.4.1 Resistive divider

A resistive divider would be the simplest circuit. It can be very precise, however, it is relatively power hungry, since it will connect resistors between the supplied voltage and ground. We can reduce this power by increasing the resistor values, however, that also increases the resistor size, requiring more chip area. Another way to reduce power is by disconnecting the divider when it is not used, by using a MOS switch.

2.4.2 Off-state MOS divider

Instead of using resistors, you can also use MOSFETs as resistors. In the off-state MOSFETs have a resistance in the GΩ range, but are much smaller than resistors with the same resistance. With equal sizing and bias, the voltage should be divided equally among the devices. However, this system is affected a lot by process variation and temperature. More devices can be connected in series to make the whole system less prone to process variation.

2.4.3 Diode stack voltage divider

Another way to divide the voltage is by a diode stack. This can be implemented using diode-connected MOSFETs. For the same sized diodes, if the same current flows through them, the voltage over them must also be the same. The diode stack divider is not affected much by temperature changes. However, the current drawn by this circuit depends strongly and non-linearly on the voltage.

2.4.4 Switched-capacitor voltage division

Like in the charge pump, we can use a switched-capacitor circuit for voltage division or multiplication. Voltage division is done by charging a stack of capacitors in series, then switching them in parallel. If the capacitors are the same value then the voltage at the output is the input voltage divided by the number of capacitors.

An advantage of this circuit is that it has no steady-state dissipation, except for leakage currents of the capacitors. The disadvantages of this circuit are that capacitors are relatively big and thus this circuit requires a large area. Also, the switching behaviour of the capacitors induces voltage spikes.

A trade-off of the methods is given in Table 2.3.

From Table 2.3 we conclude that the best circuit to use is the diode stack divider. The off-state MOS divider is inaccurate, and the switched-capacitor divider is too large and introduces voltage spikes. The resistive divider is simpler to implement, but is also quite big and has high power consumption.

Table 2.3: Voltage sensing method comparison

Method	Accuracy	Size	Power consumption
Resistive divider	High	Large	High
Off-state MOS	Low	Small	Medium
Diode stack	High	Small	Medium
Switched capacitor	Medium	Large	Low

2.5 Accuracy

The system will be integrated into the NOWI NH16, which means that the NOWI NH16 affects the accuracy of the measured illuminance. Looking at the system components, parts that can influence the accuracy are:

- Solar cell model accuracy
- Solar cell accuracy due to PVT variations
- MPP offset
- Sense resistor accuracy due to parasitics or PVT
- SenseFET accuracy, due to the voltage drop, matching, PVT, or amplifier offset
- Voltage division circuit accuracy
- Charge pump conversion ratio accuracy, due to its output resistance, capacitor matching, capacitor drift, gate-driving losses, or bottom plate losses.
- Current mirror accuracy due to matching or PVT
- ADC capacitor ladder matching
- ADC effective number of bits (ENOB)
- Reference voltage accuracy
- Ripple voltage

These effects can be divided into effects causing a voltage error, or current error. Some cause an error in both.

2.5.1 Solar cell

Solar cells suffer from PVT variation, meaning that even when your current and voltage measurements are 100% correct, it cannot be guaranteed that the measurements can be related 1-on-1 to illuminance. Much focus in research is on how temperature changes the MPP of the solar cell. With a single multiplication, constant illuminance can be related to short-circuit current I_{sc}. Via another constant, illuminance has a logarithmic relationship with the open-circuit voltage V_{oc}. However, it turns out that even though a decent estimation over a range can be made with the correct multiplication factor, the factors themselves are temperature dependent [47].

In the datasheet of solar cells temperature coefficients are given, which give the change of both I_{sc} and V_{oc} with temperature. Over a small range in the order of 20°C, this can be assumed to be linear. However, a linear coefficient does not cover the entire spectrum of the solar cell perfectly.

To estimate the error introduced by temperature variations at MPP, we can divide the temperature effect into a current error and a voltage error. The solar cell works approximately in its MPP. $I_{mpp} \approx 0.9 I_{sc}$, $V_{mpp} \approx 0.7 V_{oc}$ [48], [49]. The current and voltage errors made in the MPP due to temperature are approximately $I_{error,mpp} \approx \frac{I_{sc}}{I_{mpp}} \alpha \Delta T \approx 1.11 \alpha T$, $V_{error,mpp} \approx \frac{V_{oc}}{V_{mpp}} \beta \Delta T \approx 1.43 \beta \Delta T$. In which α is the solar cell short-circuit current temperature coefficient, and β is the open-circuit voltage temperature coefficient.

To see what the effect of temperature is on solar cells, we look at some solar cells that are often used in energy-harvesting applications, in this case, the Panasonic Amorton AM-5610 [50], and the IXOLAR KXOB25-14X1F [51].

For the Panasonic Amorton AM-5610 solar cell, α is 0.08%, and β is about 0.3%. This means to get a current error of 1% at standard test conditions (STC) there needs to be a temperature change of $\frac{0.01}{0.0008*\frac{I_{sc}}{I_{mpp}}} = \Delta T = 10.5$°C.

For a voltage error of 1%, there needs to be a temperature change of $\frac{0.01}{0.003*\frac{V_{oc}}{V_{mpp}}} = \Delta T = 2.4$°C.

Doing the same calculation for the IXOLAR KXOB25-14X1F, the temperature change needed is 20.8°C and 3.3°C for a 1% current and voltage error respectively.

A decent estimation of the maximum error due to temperature would be the maximum error on the illuminance caused by either the voltage or the current.

In case only current is used to determine the illuminance, the error calculation is simple, as then the error is linear with temperature offset.

A last effect that influences the accuracy of the solar cell is the light source. A solar cell does not absorb every wavelength equally, meaning that for a truly accurate measurement of illuminance with a solar cell, that needs to be corrected. For sunlight, an estimation of 1 lx ≈ 0.0079 W/m^2 is sometimes used.

The relation of solar cell voltage and current is proportional to both irradiance and illuminance, so only when switching light sources this will result in an error. This error is much bigger for illuminance than irradiance, because of how the spectra of a solar cell, light sources, and the human eye vary.

Every solar cell is tested under specific conditions, of which the data is available in its datasheet. This means that the data for short-circuit current, open-circuit voltage, etc, is already corrected for the specific light source used when tested. Usually a solar simulator is used to test the parameters of the solar cells, however, some cells designed specifically for indoor use will test with a different light source, like a fluorescent light. Some solar cell datasheets contain graphs linking both the illuminance and the irradiance to short-circuit current.

Because this error source is highly dependent on the specific light source, but also the specific solar cell, it is hard to find an exact number for this error. However, looking at ambient light sensors, as mentioned in Section 2.1, this error is given in datasheets to be about 10%.

A last error involving the solar cell, is the mismatch between the solar cell and the solar cell model we use to approximate the solar cell behaviour. The size of this error depends on the method we use to calculate the illuminance. This error is further investigated in Section 3.2.1.

2.5.2 MPP offset

The MPPT should make sure the system works at MPP at all times. However, the MPPT does not have infinite settings, so there always is a slight error. Any offset from the MPP directly translates into an error in measured MPP current. The accuracy depends mainly on the precision of the current and the voltage measurements. We, therefore, expect that the error due to MPP offset stays below 2.5%.

2.5.3 Sense resistor

A resistor is used to transform current into voltage, which can be measured by the ADC. This value can be linked back to current if the resistance is known. The precision of measuring current is thus limited by the precision of this resistance. In the NOWI NH16, the value of this resistance can be varied between $1.04\,\text{k}\Omega$ and $267\,\text{k}\Omega$ by switching multiple resistors in series. The switch resistance in this feature adds to the parasitic resistance and thus to the inaccuracy.

Off-chip resistors can be made with an inaccuracy of down to 0.1%. This would result in an error of 0.1% of the measured voltage over this resistor. The highest voltage measured would be 240 mV for a current of 15 mA. The resulting error then is about $15\ \mu$A.

The errors named above could be reduced by calibrating the resistor chain. Effects that cannot (passively) be calibrated against are temperature effects. The resistance of a resistor varies with temperature. The effect on SMD resistors varies between 25 and 100 ppm/°C. Resistors have a temperature coefficient, and if the coefficient is known it can be corrected for, but this would require an accurate temperature measurement and correction circuit.

On-chip resistors have lower accuracy. Especially the absolute accuracy of on-chip resistors is lower than for off-chip resistors. On-chip resistors' absolute accuracy can vary up to 20%. Trimming or calibration techniques can improve this accuracy. In [52], a calibration technique with a phase-locked loop is used and an absolute inaccuracy of less than 3% is reached. These techniques are, however, expensive. Without trimming or calibration, the

variation in on-chip resistors has a 1σ of more than 5% [53]. Utilising device diversification, meaning that different types of resistors are combined, process variation errors can be reduced, and in [53], a 1σ of about 3% is mentioned.

2.5.4 SenseFET

The senseFET structure copies the load current with a certain multiplication factor, depending on the geometrical ratio between the two MOSFETs. This ratio can be altered if needed to keep the current within certain limits. See Figure 2.3 for a circuit diagram of the senseFET structure.

The accuracy of this copied current directly influences the measurement's accuracy. In [28], it is mentioned that this senseFET structure can be inaccurate due to the mismatch between the two MOSFETs, and amplifier offset. According to NOWI NH16 documentation, the used amplifier has an offset of less than 2.5%. According to [28] the senseFET structure can have a mismatch of about 10% to 20%, depending on the ratio between the two currents. In that mismatch is a trade-off between accuracy and power efficiency. The smaller the ratio between the MOSFETs the better the accuracy of the current measurement, since the current through the senseFET will be larger. However, the current going through this MOSFET is not delivered to the load and thus results in power loss. In the case of the NOWI NH16, there are two things to also consider: firstly, the system is built around an ADC with a 0.8 V reference, and thus the current ratio is somewhat limited by this. To compensate for that, the sense resistor value can be changed to limit the voltage, but this in turn results in lower accuracy. Another thing to consider is that the NOWI NH16 can decouple the senseFET structure to save power, and thus depending on the duty cycle the extra power used in this system might not be of too much influence. The NOWI senseFET structure has an attenuation of 1, 4, 16, or 64. This makes the senseFET relatively precise but power inefficient.

Amplifier The NOWI NH16 uses an amplifier in the SenseFET structure to keep the drain voltage of the two MOSFETs the same. Since the two MOSFETs are connected to the same source and gate voltages, this amplifier keeps the drain currents of the MOSFETs the same. The current through the MOSFET in saturation can be calculated according to Equation 2.3.

$$I_{sd} = \frac{1}{2}\mu C_{ox}\frac{W}{L}(V_{sg} - |V_T|)^2(1 + \lambda V_{sd}) \tag{2.3}$$

In which μ is the charge-carrier effective mobility, C_{ox} is the MOSFET oxide capacitance per unit area, λ is the channel-length modulation parameter, V_T the MOSFET threshold voltage, V_{sg} the source-gate voltage, and V_{sd} the source-drain voltage. μ, C_{ox}, λ, and V_T depend on technology and are the same for both MOSFETs. W and L are the dimension parameters. To get the correct current division ratio L should be the same for both, while W is M times bigger. V_{sg} is the same for both MOSFETs, as the gate and source are connected to the same nodes, and the amplifier keeps V_{sd} the same for both. This amplifier can, however, have an offset that would introduce an error. In the NOWI NH16, the error caused by offset should stay within the 2.5% that the system is designed for.

2.5.5 Voltage divider

In case we add a voltage division circuit to the system, this can affect the accuracy of the system. The accuracy of a voltage divider depends on the method used for voltage division. In case it is a resistive divider, the accuracy is based on the ratio between two resistors. Following the same inaccuracy as the sense resistor of 0.1%, we get an inaccuracy of about $\sqrt{0.001^2 + 0.001^2} = 0.0014 = 0.14\%$.

2.5.6 Charge pump output resistance

The charge pump has a certain output resistance, which results in a voltage drop, causing power loss. According to the average model, as described in [54], the output resistance is defined as in Equation 2.4, with the output voltage defined as given in Equation 2.5.

$$R_{out} = \sqrt{\left(\frac{m}{f_{sw}C_t}\right)^2 + (pR_{on})^2} \tag{2.4}$$

$$V_{out} = MV_{in} - R_{out} \times I_{out} \tag{2.5}$$

In which f_{sw} is the converter switching frequency, C_t is the total flying capacitance in the converter, R_{on} is the on-resistance of the switches in the converter, p and m depend on the topology of the converter, and are further explained in [55].

The output resistance thus reduces the power delivered to the output and causes a voltage drop. The average model can be seen in Figure 2.4.

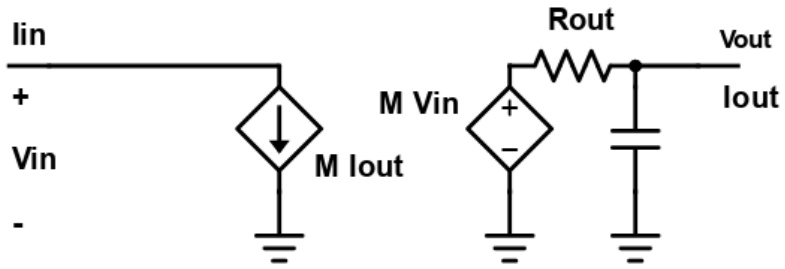

Figure 2.4: Average model diagram

The output resistance decreases if f_{sw} is increased, or if the switches are made wider, so that R_{on} decreases, however, this only saves power up to the point that gate charge losses start to dominate. Gate charge losses are estimated by $P = f_{sw}C_{gate}V_{in}^2$. In the average model, the gate charge losses are modelled by a shunt resistor at the converter input, and for measurement purposes, it simulates a current loss.

For this project, it is an option to measure the input voltage instead, in which case the voltage drop over R_{out} is not measured.

Calibration can help partly against these losses, however since the losses are not the same for all settings, there still is some variance.

2.5.7 Charge pump flying capacitors

The conversion ratio M of the charge pump depends on the ratio of the capacitors. Thus if there is any drift, component tolerances or parasitics the conversion ratio will not be as expected and thus a wrong estimation of input voltage and current will be made, resulting in an error of the estimated illuminance.

Flying capacitors have parasitic capacitances to ground since they are not connected to ground in the first place. The power lost due to these parasitic capacitances is given in Equation 2.6.

$$P_{bp} = f_{sw}V_{in}^2 \sum_{i=1}^{2n_c} |V_{nodes1,i} - V_{nodes2,i}|^2 C_{par,i} \tag{2.6}$$

In Equation 2.6, $|V_{nodes1,i} - V_{nodes2,i}|$ is the voltage swing on each capacitor node [55]. The parasitic capacitance usually is about 0.2-1.5% of the total capacitance. In the average model, these losses are modelled with a shunt resistor at the input side. The effect of these losses can mostly be seen in a lower output current. The current loss can be seen in Equation 2.7.

$$I_{bpl} = V_{in}f_{sw} \sum_{i=1}^{2n_c} |V_{nodes1,i} - V_{nodes2,i}|^2 C_{par,i} \tag{2.7}$$

Thus the effect of this depends on the switching frequency, and input voltage.

Some current of I_{in} flows into R_{bp}, the resistance simulating the bottom plate capacitor losses. This causes I_{out} to go down. According to Equation 2.8, this means that either V_{in}, V_{out}, M, or R_{out} changes.

$$I_{out} = \frac{I_{in} - V_{in}R_{bp}}{M} = \frac{MV_{in}V_{out}}{R_{out}} \tag{2.8}$$

A diagram of the average model with R_{bp} can be seen in Figure 2.5.

V_{out} is fixed by the battery, R_{out} does not change due to the addition of R_{bp}, as can be concluded from Equation 2.4, M is based only on topology. Thus this means that for Equation 2.8 to be satisfied, V_{in} needs to decrease, or I_{in} needs to increase, or in the case of a power supply like a solar cell, both happen at the same time to find a balance between I_{in} and V_{in} that satisfies Equation 2.8, and the solar cell I-V curve.

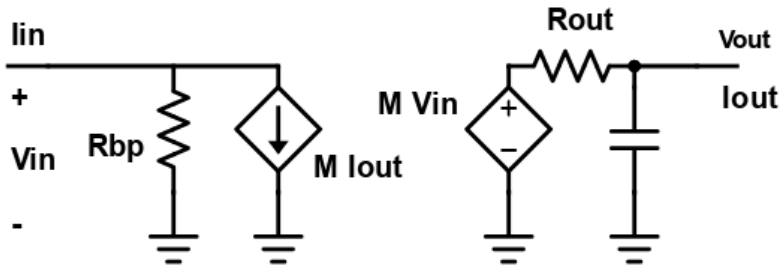

Figure 2.5: Average model diagram with R_{bp}

Even though the addition of R_{bp} is essentially a power loss, for sensing purposes it means the voltage can still be measured correctly, while there is an inaccuracy in the measured current according to $\frac{V_{in}}{R_{bp}}$ (input-referred), or $V_{ADC,error} = \frac{V_{in}}{R_{bp} \times M \times N} \times R_{sense}$

According to [56], IC capacitors can have a tolerance down to 0.1%, with the correct layout and if they are big enough. The capacitors have a temperature coefficient of about 30 ppm$/^{\circ}$C. The temperature coefficient seems negligible for the purpose of this project.

If the voltage and the current are measured at the input of the converter, then this effect can be ignored.

Looking at the NOWI NH16, the charge pump has an overall efficiency of about 90% to 95%. This is the efficiency to expect for a large range, however, some specific settings and inputs have lower accuracy.

2.5.8 Current mirror

The current mirror should copy the current to be measured as precisely as possible and pass it through the sense resistor. Any error directly affects the measured current. The main accuracy limitation of a current mirror is the MOSFET matching. If the MOSFETs have different aspect ratios, the currents will not be equal. If there is a temperature difference between the MOSFETs then this will also result in a current error. According to NOWI documentation, the current mirror has an inaccuracy of 2.5%. To reach this the current mirror has cascoding MOSFETs and degeneration resistors, which help raise the output impedance of the current mirror.

In [57], an inaccuracy of 2% is mentioned for their current mirror. In [58], a regulated cascode current mirror is used, which increases speed compared to the regular cascode current mirror. The used feedback loop keeps the mirror drain-source voltage equal, essentially increasing the output resistance. An inaccuracy of 0.5% is mentioned.

2.5.9 MOS switch on-resistance

A switch in the current path also has some resistance. If designed correctly it is expected that this resistance is in the mΩ range, and thus is much smaller than the output resistance of the converter. We, therefore, expect that this resistance has a negligible effect on the system's accuracy. If needed, this resistance can be modelled by adding this resistance to the SCPC output resistance. If the on-resistance is too large, it can be decreased by increasing the switch size.

2.5.10 Gate-driver losses

The switches in the converter require energy to open. This energy is partially drawn from the converter output, thus resulting in a current error. The power required to drive the switches is $P_{sw} = f_{sw} V_{in}^2 C$, in which C is the switches gate capacitance. Not all power drawn results in a current error, as most of it is supplied by the battery. It is therefore difficult to give a good estimation of how big of an error this effect results in.

2.5.11 ADC

The ADC turns the analog voltage into a digital signal. The ADC is supplied with a 0.8 V reference and has 8 bits. That would give the system an accuracy of $\frac{0.8}{2^8} = 3.125$mV. However, the ADC can also have inaccuracies and

according to NOWI NH16 documentation, the final ADC will have an ENOB of 6, which would mean the ADC can detect differences of $\frac{0.8}{2^6} = 12.5$ mV.

The largest current to detect within this system according to the NOWI specifications is 16 mA. This current should correspond to a voltage of 0.8 V. Given the maximum error of 12.5 mV, the worst-case error in current caused by only the ADC would be up to $16 \times \frac{12.5}{0.8} = 25$ μA, resulting in an error of about 1.6%.

Every ADC has a trade-off between power, accuracy and speed. Speed is set within the system. Speed can be reduced if battery voltage gets too low, saving power. A trade-off between power and accuracy is left.

The physical limit of a SAR ADC is the accuracy of the DAC capacitor ladder. If better accuracy is required, then the size of the capacitors can be increased, which makes the DAC more precise. However, this comes at the cost of more area and power. Another option is to increase the number of bits, which also comes at the cost of more area and power.

In [59], a SAR assisted digital slope ADC is used. A figure of merit (FoM) of 2.63 fJ/conversion-step is mentioned. It has an ENOB of 10.4, resulting in an error of 0.075%. In [60], an asynchronous SAR ADC is used with an ENOB of 10.85 and a 0.9 V supply voltage, resulting in an error of 0.5 mV, or an inaccuracy of 0.056%.

2.5.12 Reference voltage

The NOWI NH16 uses a 0.8 V reference voltage in its ADC. There is an accuracy limit to this depending on how this voltage is created. The bandgap reference created is based on [61], which mentions a 3σ of 2.9%, which is more or less the inaccuracy of the reference. Better bandgap references exist. The main accuracy limit is process variations and can have a 3σ inaccuracy of 1.25% [62].

2.5.13 Ripple voltage

An SCPC has a certain output voltage ripple resulting in a current ripple as well. The output voltage ripple of an SCPC is: $\Delta V_o = \frac{I_{out}}{f_{sw}(C_{out}+C_1)}$, in which C_{out} is the total capacitance at the output node. In the NOWI NH16, a capacitor is placed at the output to filter this ripple. Another option to decrease the ripple voltage, is to increase the switching frequency. However, this has direct consequences for the output impedance, which in turn results in different loading conditions of the solar cell. This causes the solar cell to drift away from its MPP. The capacitor together with the output resistance of the converter also causes a delay according to $\tau = R \times C$, so a larger capacitor also causes a bigger delay in the system. Thus there is a trade-off between ripple voltage and delay, in which there is a maximum value of the capacitor, as to not exceed a certain delay.

In the NOWI NH16, there are two capacitors that can be used to filter the ripple voltage. This is to be able to tune the system over a larger range of output resistance, and thus a larger range of switching frequencies. The input capacitor together with the MOSFET switch resistance, R_s, form a low-pass filter. The low-pass filter is supposed to filter out ripples generated by the charge pump due to switching. The larger this capacitance the better it filters the ripple, according to $I_{rip} = \frac{I_{avg}}{f_{sw}R_{s1}C_{mppt}}$. The NOWI NH16 allows for an output ripple of 30%, which is filtered by the amplifier in the current sensing path to under 2.5%. The latter suggests a trade-off between ripple voltage and bandwidth. Any variance in the capacitance value should not have too big of an effect on the sensing accuracy.

2.6 Accuracy effect

The goal of the project is to calculate the illuminance, G. The effects of components on the accuracy of the system influences the calculated illuminance. To get an overview of what effects are important to consider, we need to know how much influence each effect has on G. Effects can influence either the calculated voltage, current, or both. So to get a full overview, it is important to consider the effects of the voltage (v) and current (i) on G. According to the four-parameter model, further described in Section 3.2.1, G can be calculated if i and v are known according to Equation 2.9 [63]. In this equation, we see that the illuminance calculation depends on the sum of one term influenced only by the current, and another with an exponential relation to both the current and the voltage.

$$G = \frac{G_0 i}{I_{sc}[1 + \alpha(T - T_0)]} + G_0 e^{\frac{v + R_s i - V_{oc} - \beta(T - T_0)}{AV_t n_s}} \tag{2.9}$$

To compare the accuracy effect we use a common unit to translate all the accuracy effects to, namely the responsivity. The responsivity of the sensor is defined in lx/least significant effective bit (LSEB), as calculated by $\frac{V/LSEB}{V/lx}$, or $\frac{I/LSEB}{I/lx}$. The responsivity thus depends on several factors, namely:

- Solar cell type, including size
- ADC ENOB
- Dynamic range

The responsivity of the sensor depends a lot on the solar cell because a bigger solar cell means a higher current and/or voltage for the same irradiation. This means higher voltage or current per lux and thus a better responsivity.

The other part that will affect the responsivity is the ADC ENOB. The only way to improve this is to improve the ADC, as this would improve the V or I per LSEB.

Another way to improve V or I per LSEB is to have more options for the value of the sense resistor. In that case for a lower current, we use a larger resistor, which in turn means the input voltage for the ADC will be larger. The ADC will then have a higher output, thus increasing V or I per LSEB. The NOWI NH16 already uses this principle, and it has four different ranges.

If we know the responsivity we can translate each error found in this chapter, in either current or voltage, to its resulting error in lux.

A summary of the accuracy limitations can be found in Table 2.4.

Table 2.4: Accuracy limits summary

Component	**Main accuracy limitations**	**Inaccuracy**	**State of the art inaccuracy**	**Calibration possible**
Solar cell	Light source variation	10%	N.A.	No
Solar cell	Model mismatch	5%	N.A.	Yes
Solar cell	Temperature	3%	N.A.	No
MPPT	MPP offset	2.5%	$<$2% [49]	No
Sense resistor	Tolerance, parasitics, thermal drift	20% (absolute) 2.5% (relative)	3% [52] [53] 0.1% (SMD) [25]	Yes
SenseFET accuracy	Mismatch, thermal drift, resistance	2.5%	1% [27]	No
Voltage division circuit	Tolerance, parasitics, thermal drift	2.5%	0.14% [25]	Yes
Charge pump conversion ratio and power loss	Tolerance, parasitics, capacitor ESR	5-10%	8% [64]	Partly
Charge pump: bottom plate losses	Parasitics	<1.5%	0.2-1.5% [55]	No
Charge pump: conversion ratio	Capacitor tolerance	2.5%	0.1% tolerance per capacitor [56]	No
Charge pump: Gate driving losses	Power used to turn on MOSFETs	<1%	N.A.	No
Current mirror	PVT	2.5%	0.5% [58]	No
ADC ratio	Capacitor ladder matching	2.5%	0.1% tolerance per capacitor [56]	No
ADC ENOB	Capacitor ladder size	1.6%	>0.1% [60]	No
Reference voltage	Resistor and MOSFET matching	2.9% [61]	1.25% [62]	Yes
Ripple voltage	Trade-off with bandwidth	2.5%	N.A.	No

2.7 Conclusion

We concluded that though similar works exist, not one of them harvest energy as their primary function, and with the same solar cell sense illuminance.

We have given a breakdown of sensing methods, and we concluded that the senseFET method, as the NOWI NH16 uses, is the best method to do current sensing for this project.

Looking at Table 2.4, we see that most errors are around 2.5%, which corresponds to the point the NOWI NH16 is designed around. The biggest error is in the sense resistors absolute accuracy, which is 20%. However, the system will be calibrated for this.

Other than that we can see that the biggest error to expect is up to 10% from charge pump conversion efficiency. This error can partly be calibrated for. We expect, however, that even after calibration some error is left due to charge pump conversion efficiency variations.

Other significant errors come from the variation and temperature differences in the solar cell. This causes a modelling error of about 5%, and temperature can cause another error of about 3%. Lastly, looking at ambient light sensors an inaccuracy limit of about 10% is given, to account for light source variation, caused by the difference in spectral properties of the eye compared to solar cells.

In conclusion, for a reasonable range, we expect that the error in our system is around 10%.

3.
System design

In this chapter, we discuss four different options for the top-level system design and make a trade-off between them. The designs differ in what quantities need to be measured to make an illuminance estimation. Design 1 measures both MPP current and voltage, Design 2 measures short-circuit current, Design 3 measures open-circuit voltage, and Design 4 measures only MPP current.

In the end, we make a trade-off and choose a design. The hardware implementation for that design will be discussed in Chapter 4.

3.1 Prerequisites

The designs have different results with different environmental conditions, like illuminance and temperature. Therefore it is important to find what conditions make sense, to narrow down the possible results and get a good idea of how each design performs under realistic conditions.

3.1.1 Reference solar cells

To get an idea of how well the different designs perform, we simulate them. Since many variables influence the final result, we choose a logical set of parameters. Part of these parameters are the solar cell characteristics. Therefore we choose some standard solar cells that are often used for energy harvesting. The solar cells we use as a reference are the IXOLAR KXOB25-14X1F [51] and the Panasonic Amorton AM-5610 [50], specialized for outdoor use, and the Panasonic Amorton AM-1456 [65], specialized for indoor use. These solar cells have been determined by the NOWI application team to be the best solar cells to use in combination with the NOWI NH16. We also use the last two solar cells for getting input data for system validation.

3.1.2 Illuminance ranges

We simulate the performance of our design at several points of illuminance to check how well the performance is over the entire spectrum. We choose the points based on environmental conditions that the sensor might encounter. These are:

- Living room: 50 lx
- Office space: 400 lx
- Outside cloudy: 1000 lx
- Indirect sunlight: $10,000$ lx
- Direct sunlight: $100,000$ lx

3.1.3 Calibration

We calibrate the system. This is necessary because, firstly, the NOWI NH16 has a relative inaccuracy of about 2.5%, but an absolute inaccuracy of about 20%, which is not good enough. This accuracy limit is caused by a reference resistor whose inaccuracy can vary about 20% between chips. This is not a problem for the MPPT since it only checks one power relative to the previous one, but for this project, there should be an absolute number of illuminance at the end.

The additional advantage of calibrating is that the system can then be calibrated for a specific solar cell, as all solar cells are slightly different, and thus would give a different result without calibration. Then the system can also be calibrated around a specific temperature. In the datasheet, the temperature is usually 25°C, however in reality, indoors or outdoors the temperature is mostly lower.

The way of calibrating depends on the exact design, however, they all work approximately the same. All systems will consist of the following components:

- Way to input a calibration reference value (digitally)
- Comparing block, to compare the reference value with a measured and computed value
- Way to calculate the needed value for the calibration parameter
- Control FSM, to control the calibration process

We have to alter the formula to calculate illuminance from measured parameters to include a calibration parameter, which we calibrate so that the system's output matches a reference input. For Designs 2, 3, and 4, calibration can be done with a single parameter. Design 1 depends on both current and voltage, and thus for perfect calibration, two calibration parameters are necessary. A reasonable calibration can, however, be done using only a single parameter.

3.2 Design 1: MPPT based design, using both voltage and current

The first design is based on keeping the MPPT working at MPP at all times. This means that we do current and/or voltage measurements at MPP, and with that information, we make an estimation of the illuminance. The advantage is that this way no energy is lost while doing measurements, and there is minimal impact on the MPPT. A disadvantage is that when working at, or near, the MPP, it is harder to make a good estimation of illuminance. The relation with illuminance at MPP includes an exponential part, as described later on in Equations 3.1 and 3.6. The exponential part has high computational complexity, making the system bigger and more power hungry.

A block diagram of the design can be seen in Figure 3.1.

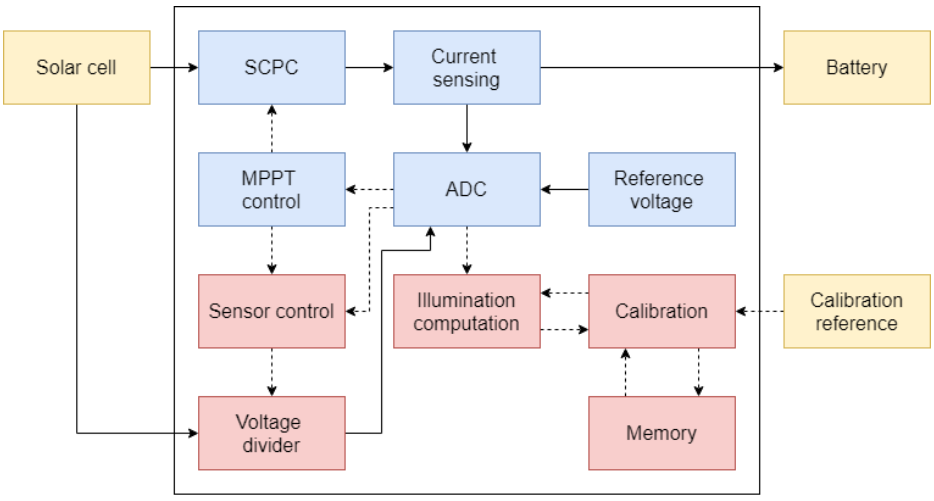

Figure 3.1: Design 1 block diagram

In Figure 3.1 parts already available in the NOWI NH16 are given in blue, parts that would need to be designed are given in red. Everything in yellow is connected externally to the chip. A solid line shows an analog signal, while a dashed line shows a digital signal.

3.2.1 Solar cell model

Working at MPP means that we must make a translation between current and/or voltage at MPP and illuminance. Therefore we need a good model describing the solar cell, linking current and/or voltage to illuminance.

Five-parameter model

Simulation of the system we design requires a good model of the energy harvester, in this case, a solar cell. An often-used model for a solar cell can be seen in Figure 3.2. The parameters of this model can be derived using only values given in the solar cell datasheet, as first given in [66]. A good mathematical breakdown of the five-parameter model is given in [67], and has been expanded upon in [68].

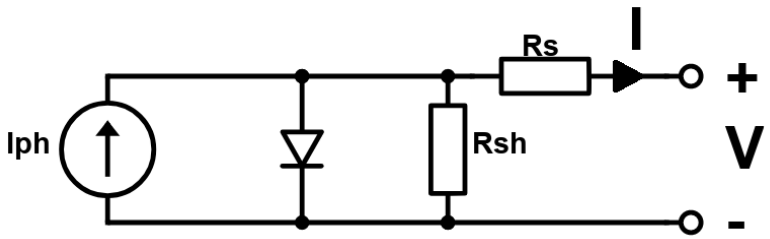

Figure 3.2: Five-parameter solar cell simulation model

The five-parameter model gives a current-voltage relationship of a solar cell as:

$$i = I_{ph} - I_o \left(e^{\frac{v+iR_s}{n_s A V_t}} - 1 \right) - \frac{v + iR_s}{R_{sh}} \tag{3.1}$$

in which

$$I_o = I_{sc} e^{-\frac{V_{oc}}{n_s A V_t}} \tag{3.2}$$

and

$$I_{ph} = I_o e^{\frac{V_{oc}}{n_s A V_t}} \tag{3.3}$$

In Equation 3.1, V_t is the thermal voltage which is given as $V_t = \frac{kT}{q}$, in which k is Boltzmann's constant, q is the elementary charge, and T is the temperature. A is the diode ideality factor. n_s is the number of cells in series. I_{ph} is the photo-generated current, meaning it is the physical current generated by light, before non-idealities. I_o is the dark saturation current. I_{sc} is the short-circuit current.

We have made a MATLAB script to implement the calculations as done in [68], to calculate R_s, R_{sh}, and A. We then fill in these parameters into Equation 3.1 to draw an I-V curve. We also made a MATLAB script to draw these curves. We then use the simulated I-V curve data to simulate any solar cell that may be used with the system. The MATLAB files can be found in Appendix G.

Four-parameter model

A disadvantage of the previous model is that the parameters cannot be derived directly. A simplification that we can make is by assuming the shunt resistance is infinite. The simplified circuit can be seen in Figure 3.3.

In the four-parameter model, only A and R_s need to be calculated, which we can do directly as opposed to the five-parameter model [63]:

$$V_t A n_s = \frac{(2V_{mpp} - V_{oc})(I_{sc} - I_{mpp})}{I_{mpp} + (I_{sc} - I_{mpp}) \ln(1 - \frac{I_{mpp}}{I_{sc}})} \tag{3.4}$$

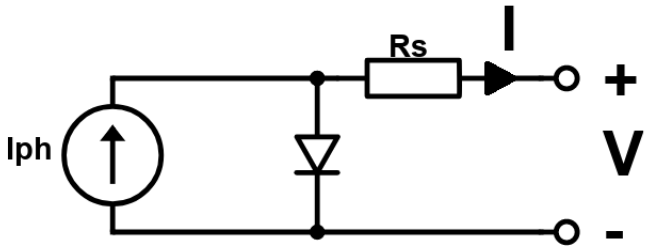

Figure 3.3: Four-parameter solar cell simulation model

$$R_s = \frac{V_{mpp}}{I_{mpp}} - \frac{2V_{mpp} - V_{oc}}{I_{mpp} + (I_{sc} - I_{mpp})\ln(1 - \frac{I_{mpp}}{I_{sc}})} \tag{3.5}$$

The I-V curve can then be calculated using Equation 3.6.

$$i = I_{ph} - I_o \left(e^{\frac{v+iR_s}{n_s V_t}} - 1 \right) \tag{3.6}$$

Model relation to physical quantities

Since the goal is to calculate the illuminance from the four- and five-parameter models, it is important to know how the illuminance relates to the equations given above. These equations are also temperature dependent, and thus we also explore that relation.

The dependencies related to illuminance, G, or temperature, T, are as follows:

V_t has a temperature dependency, as $V_t = \frac{kT}{q}$. With changing temperature, to calculate V_t from V_{t0}, Equation 3.7 can be used.

$$V_t = \frac{T}{T_0} V_{t0} \tag{3.7}$$

The open circuit voltage, V_{oc}, has both a temperature and an illuminance dependence as given in Equation 3.8.

$$V_{oc} = V_{oc0}(1 + \beta(T - T_0) + V_t A n_s \ln\left(\frac{G}{G0}\right) \tag{3.8}$$

The short-circuit current, I_{sc}, has a temperature and illuminance dependence according to Equation 3.9.

$$I_{sc} = \frac{G}{G_0} I_{sc0}(1 + \alpha(T - T_0)) \tag{3.9}$$

In these equations V_{t0}, V_{oc0}, and I_{sc0} represent V_t, V_{oc}, and I_{sc} under standard test conditions (STC) respectively, so at G_0 and T_0. What the standard test conditions are, is mentioned in the datasheet, however, choosing any other point for G_0 and T_0 is also valid.

Model accuracy

According to [69], the inaccuracy of the five-parameter model around the MPP is about 5%. It should be noted that the type of solar cell used in this project is very different from the one used in [69], so the inaccuracy of the model to the cells used for our application might be different.

Calculating the voltage and current error at the MPP of the two curves given in the Panasonic Amorton AM-5610 [50], the Panasonic Amorton AM-1456 [65], and the IXOLAR KXOB25-14X1F [51] datasheets, for the four-parameter model gives the errors given in Table 3.1.

Looking at the errors, we conclude that the model is sufficienty accurate at STC, around the MPP, with an error that stays under 2%. However, further away from STC there is a significant drop-off of the accuracy, with errors of around 10%.

We can do the same for the five-parameter model. The errors are given in Table 3.2.

The simulated errors are very similar to the errors found in the four-parameter model. The conclusion, therefore, is the same. The model is good around STC, but not too good at different conditions.

Table 3.1: Four-parameter model vs datasheet error

	I error	**V error**
Panasonic	$86.7\ \mu$A	63.8 mV
1000 W/m^2	1.9%	1.6%
Panasonic	$191\ \mu$A	397 mV
440 W/m^2	9.4%	10.1%
IXOLAR	$800\ \mu$A	5.7 mV
1000 W/m^2	1.5%	1.0%

Table 3.2: Five-parameter model vs datasheet error

	I **error**	V **error**
Panasonic	$75.8\ \mu$A	58.2 mV
1000 W/m^2	1.6%	1.5%
Panasonic	$252\ \mu$A	370 mV
440 W/m^2	13.1%	9.2%
IXOLAR	1.1 mA	7.7 mV
1000 W/m^2	2.0%	1.4%

In short, this means that when using Design 1 for a larger illuminance range, the inaccuracy will not be lower than 10%.

3.2.2 Calculating illuminance

We can use the four- and five-parameter models to reverse engineer the illuminance. The I-V curve is dependent on two factors, the illuminance and the temperature. For now, we do not consider the temperature, since measuring temperature requires extra hardware.

Reverse engineering the illuminance depends on the model. In [70] three model-based methods for calculating illuminance from measurements are described. The main formula used to calculate illuminance is based on the five-parameter model and is Equation 3.10.

$$G = \frac{i + i_{d0}[\frac{T}{T_0}]^3 \text{exp}(\frac{E_{g0}}{kT_0} - \frac{E_g}{kT})[\text{exp}(q\frac{v+iR_{s0}n_s}{AkTn_s}) - 1]}{\frac{1}{G_0}[(I_{ph0} + \alpha(T - T_0)) - \frac{v+iR_{s0}n_s}{R_p n_s}]} \tag{3.10}$$

i_d is the diode current, as given in Figure 3.2.
E_g is the bandgap energy, which in other formulas is assumed constant, but in [70] it is temperature-dependent.
A subscript 0 at any parameter, indicates it is the parameter at STC.

Based on the four-parameter model, a relatively simple equation for the illuminance can be found. [71] gives the illuminance as given in Equation 3.11.

$$G = \frac{G_0 i}{I_{sc}[1 + \alpha(T - T_0)]} + G_0 e^{\frac{v+R_s i - V_{oc} - \beta(T-T_0)}{AV_t n_s}} \tag{3.11}$$

In which it is assumed that $e^{\frac{V_{oc}}{V_T}} >> 1$ and $V_T \approx V_{T0}$.

MPP linearisation An idea is to linearise the formula around the MPP, which makes the system less computationally complex, as it removes the exponent from Equation 3.11. This is possible, because the system is connected to an MPPT and thus it is expected that the solar cell works at, or close to, its MPP. We did not use this idea in the end, however. A description of the linearisation idea can be found in Appendix A.

3.2.3 Input or output voltage sensing

In Design 1, we estimate the voltage as generated by the solar cell. We do this by measuring the voltage and calculating what the input voltage would be. Two logical positions for the voltage measurement are either the

output voltage of the SCPC or the input voltage directly at the solar cell.

Measuring the output voltage has several advantages. Firstly, this way we can use the measured voltage to calculate the output power, which is interesting to know for other applications of the NOWI NH16. Also, the output voltage is connected to a battery and thus stays relatively constant. This means that these voltage measurements can be infrequent, and since every measurement costs energy this is a useful property. On the other hand, since we need to know the solar cell voltage, measuring the voltage at the output requires us to calculate the input voltage by dividing the output voltage by the charge-pump conversion ratio.

There are other options. Since the voltage measurement is used to get insight into the solar cell voltage, it makes sense to measure the voltage at the input, directly where the solar cell is connected. This eliminates the SCPC output impedance and current mirror switch resistance from the path, which both introduce a voltage drop.

Also, the conversion ratio of the charge pump would not be included in this measurement. By measuring the input voltage instead of the output voltage the equation obtained to calculate the solar cell voltage becomes Equation 3.12.

$$v = \frac{R_1 + R_2}{R_1} V_{ADC} \tag{3.12}$$

In Equation 3.12, R_1 and R_2 are the resistors that would be used in the voltage division. The only thing left in the path is a voltage division, as the ADC can only handle voltages under $0.8\,\mathrm{V}$. For more precise measurements, measuring the input voltage is preferred.

3.2.4 Design limits

The accuracy of Design 1 is limited by:

- Model accuracy
- Temperature
- The accuracy of the voltage-sensing circuit
- The accuracy of the current-sensing circuit
- The accuracy of the ADC
- The accuracy of the calibration circuit

Model accuracy

The biggest source of error for Design 1 is in the model accuracy. The models are designed to fit well around the parameters as given in the datasheet, which makes them accurate around the MPP at STC. However, the further away conditions get from STC, the worse this error gets, as illustrated in Tables 3.1 and 3.2. For a usable range, the error the model introduces is more than 10%.

Temperature

In Design 1, both the short-circuit current temperature parameter, α, and the open-circuit voltage temperature parameter, β, affect the system. The effect of the two, however, is opposite, meaning the error caused by β is partially compensated by α. A simple estimation would be to assume the error is the worst case of the two, usually β as it is much larger than α. Simulation shows that for the Panasonic Amorton AM-5610 around STC the error is about $0.3\%/°\mathrm{C}$.

Voltage-sensing circuit

For voltage sensing, we can use the same ADC as for current sensing. The ADC has a reference voltage of $0.8\,\mathrm{V}$, so if the solar cell voltage exceeds this, which it probably will, a voltage-division circuit is needed. Since the ADCs inaccuracy is 2.5%, the voltage-sensing circuit does not need to be much more accurate than this. Any of the voltage-division circuits as mentioned in Chapter 2.4 should suffice.

Current-sensing circuit

The NOWI NH16 already has a current-sensing circuit that is used for the MPPT. The system has an inaccuracy of 2.5%.

ADC circuit

The ADC in the NOWI NH16 has an inaccuracy of about 2.5%, with an ENOB of about 6. In Design 1, we use the ADC for both sensing the current and the voltage, and with that this is probably the accuracy limit for both.

Calibration circuit

Since the system will be calibrated, the accuracy of this calibration plays a role in the accuracy of the design. The calibration uses the ADC to compare, so the ADCs accuracy is also the accuracy limit of the calibration circuit.

3.2.5 Accuracy simulation

The responsivity of this model is harder to determine, as both current and voltage influence the estimated illuminance. To get a sense of responsivity we look at the Panasonic Amorton AM-5610 again. For this solar cell at STC, the responsivity is 24.9 klx$/$mA and 402 klx$/$V if used over the full range. It is, however, possible to split the current and voltage measurements in multiple ranges for the ADC. Meaning that if the current or voltage is low, you do not divide them by the same factor. This helps to use the ADC more efficiently, improving the responsivity for lower inputs. In theory, if you make enough different division ranges, the limit becomes the ENOB of the ADC.

Without including the model error, we estimated an error in the measured current of 2.5% and also an error in the measured voltage of 2.5%. Then with the four-parameter model, we simulate the Panasonic Amorton AM-5610 and we check what the error in lux is when there is a voltage and current error of 2.5%. Over the range, as described in Chapter 3.1.2, the MATLAB simulation shows that the error in lux is up to 5%.

However, the model error causes a current and voltage error of about 10% in this range. Therefore, we repeat the simulation with this error. The error in lux made for Design 1 can be up to 35%, see Table 3.4.

We repeat the simulations for the Panasonic Amorton AM-1455, an indoor solar cell. The simulated error can be found in Table 3.3, however, for this solar cell even for an error of 2.5% the error in lux is too big to be usable.

Table 3.3: Panasonic Amorton AM-1456 simulated error for Design 1

Range type	**Illuminance (lx)**	I_{mpp} **(**μ**A)**	V_{mpp} **(V)**	**Error (2.5%)**
Living room	50	1.2	1.77	3 lx
Office space	400	9.6	1.30	62 lx
Cloudy	1000	24	1.31	390 lx
Indirect sun	10,000	240*	1.51	7100 lx
Direct sun	100,000	4000*	2.14	84,000 lx

Values with an asterisk (*) fall outside the usable range as given in the datasheet.

Table 3.4: Panasonic Amorton AM-5610 simulated error for Design 1

Range type	**Illuminance (lx)**	**Error (2.5%)**	**Error (10%)**
Living room	50	2 lx	5 lx
Office space	400	12 lx	53 lx
Cloudy	1000	35 lx	140 lx
Indirect sun	10,000	410 lx	1800 lx
Direct sun	100,000	5300 lx	34,000 lx

3.2.6 Conclusion

Design 1 has as advantage that it works at MPP, and thus has minimal impact on the NOWI NH16. However, simulations show that the design is not very accurate, and probably not accurate enough for use in this project.

3.3 Design 2: Short-circuit current based

Design 2 focuses on accuracy. The difference with Design 1 is that we use the short-circuit current, I_{sc}, to determine illuminance. The advantages are that measuring the voltage is not necessary, since at short-circuit current the input voltage is zero. This also simplifies the solar cell model, since if $V = 0$ the illuminance is linear with I_{sc}. Thus, no complex calculations are needed, and the model is more accurate. We can solve errors due to solar cell variations by calibration.

The formula that is left in short-circuit conditions is given in Equation 3.13.

$$G = \frac{G_0 I_{sc}}{I_{sc0}[1 + \alpha(T - T_0)]} \tag{3.13}$$

When ignoring temperature differences, Equation 3.14 is left.

$$G = \frac{G_0 I_{sc}}{I_{sc0}} \tag{3.14}$$

If the temperature is not corrected for, we can still calibrate the system around an operation point of our choosing, making the design accurate around this specific temperature point. In the case of this system, we only need to alter the factor $\frac{G_0}{I_{sc0}}$ to $\frac{G_0}{I_{sc0}[1+\alpha(T_{cal}-T_0)]}$, which when multiplied by the measured short-circuit current gives the illuminance, as in Equation 3.13.

A disadvantage is that for the system to measure the short-circuit current, we have to short-circuit the solar cell, which means the converter has to be turned off. During this time no energy is harvested, and it could cause transient effects in the converter. Also, it is still necessary for the MPPT to measure the output current, and thus, two current-sensing systems are required.

A block diagram of the design can be seen in Figure 3.4.

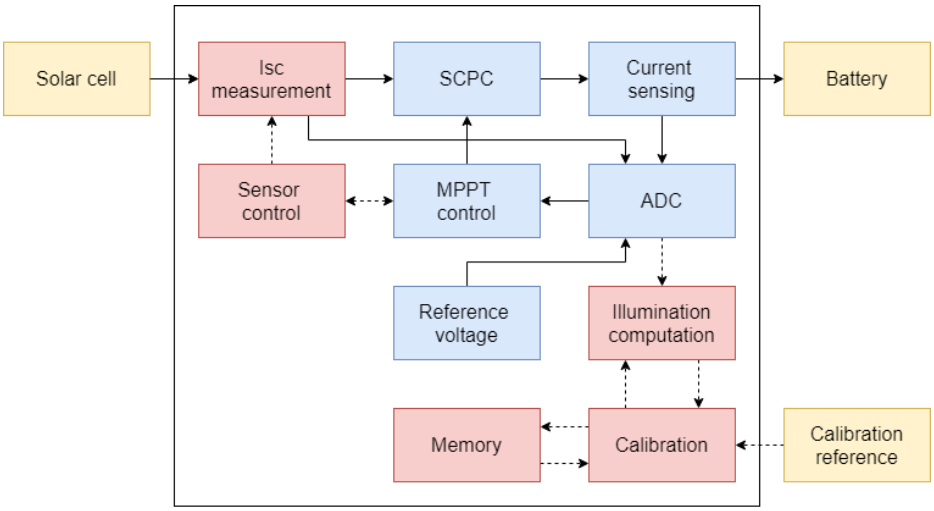

Figure 3.4: Design 2 block diagram

3.3.1 Calibration

Since the absolute accuracy of the NOWI NH16 is not good enough, the system will be calibrated. This will also help against any gain errors in the system.

We use Equation 3.14 to calculate the illuminance. To calibrate, we add a parameter C as in Equation 3.15.

$$G = \frac{C \times G_0 \times I_{sc}}{I_{sc0}} \tag{3.15}$$

3.3.2 Design limits

The short-circuit current design is limited by:

- Temperature
- The accuracy of the current-sensing circuit
- The accuracy of the ADC
- The accuracy of the calibration circuit

Temperature

The short-circuit current has a dependency on temperature, but it is relatively small, about $0.1\%/^{\circ}$C for the Panasonic Amorton series.

Current-sensing circuit

Besides the ADC, the current-sensing circuit could also be the limit of Design 2. The NOWI NH16 has an inaccuracy of about 2.5% in its current-sensing circuit. For Design 2, we need to design another current-sensing circuit, and with similar accuracy to not degrade the accuracy of the system.

ADC accuracy

In the NOWI NH16, the current is sensed with an inaccuracy of about 2.5%. For the full range of 0 to I_{sc}, this would mean a responsivity of 21 klx$/$mA for the Panasonic Amorton-5610 [50], resulting in 5.2 klx$/$LSEB. However, by dividing the current, we can divide this range, improving the responsivity. In the NOWI NH16, the current mirror already divides the current to suit the ADC. Using those ranges, the smallest range goes from 0 to 4 μA. Within this range, the responsivity is 1.3 klx$/$LSEB.

Calibration circuit

We will calibrate the system via a calibration circuit, but as in the previous design, the ADC accuracy is the limit in the calibration circuit.

3.3.3 Accuracy simulation

Like for the previous design, we simulate the error of Design 2, when there is a current error of 2.5%. The results can be seen in Tables 3.5 and 3.6 for the indoor and outdoor solar cells respectively. Since the illuminance is linear with the short-circuit current, the error is also linear. This means that the error of 2.5% as made by the ADC and current measuring system results in a small error in the illuminance. Thus, we can reach a decent accuracy. Taking into account the biggest error of 10%, and considering Design 2 has a linear relation, an error of 10% in short-circuit current causes an error of 10% in the illuminance.

Table 3.5: Panasonic Amorton AM-1456 simulated error for Design 2

Range type	Illuminance (lx)	I_{sc} (μA)	Error (2.5%)	Error (10%)
Living room	50	1.5	1 lx	5 lx
Office space	400	12	10 lx	40 lx
Cloudy	1000	30	25 lx	100 lx
Indirect sun	10,000	300*	250 lx	1000 lx
Direct sun	100,000	3000*	2500 lx	10,000 lx

3.3.4 Conclusion

Design 2 has good accuracy and is, therefore, a reasonable option to choose. The disadvantage is that this design has a big impact on the design and functioning of the MPPT.

Table 3.6: Panasonic Amorton AM-5610 simulated error for Design 2

Range type	Illuminance (lx)	I_{sc} (mA)	Error (2.5%)	Error (10%)
Living room	50	0.0024	1 lx	5 lx
Office space	400	0.019	10 lx	40 lx
Cloudy	1000	0.048	25 lx	100 lx
Indirect sun	10,000	0.48	250 lx	1000 lx
Direct sun	100,000	4.8	2500 lx	10,000 lx

3.4 Design 3: open-circuit voltage based

Design 3 uses the open-circuit voltage instead of the short-circuit current to determine illuminance. The main advantage of this compared to Design 2 is that there is a moment during MPP tracking when the input impedance is high enough to assume $V_{meas} \approx V_{oc}$. This moment can be used to measure the input voltage, which is then close to the open-circuit voltage. This way the converter does not have to turn off at any point.

At open-circuit, the solar cell can be described by Equation 3.16 [71]. This equation assumes the four-parameter model, with $e^{\frac{V_{oc}}{V_T}} >> 1$, and $V_T \approx V_{T0}$.

$$V_{oc}(G) = V_{oc0} + n_s A V_{T0} \ln\left(\frac{G}{G0}\right) \tag{3.16}$$

Rewriting Equation 3.16, gives that the illuminance can be calculated according to Equation 3.17.

$$G = G_0 e^{\frac{V_{ADC} - V_{oc0}}{n_s A V_{T0}}} \tag{3.17}$$

According to [67] Equation 3.18 can be used to calculate V_{oc}. This equation is based on the five-parameter model and thus is without the simplifications that Equation 3.16 assumes. Equation 3.18 does require an iterative calculation to solve.

$$V_{oc}(G) = \ln\left(\frac{\frac{G}{G_0} I_{ph} R_{sh} - V_{oc}(G)}{I_o R_{sh}}\right) n_s A V_t \tag{3.18}$$

The datasheet from the Panasonic Amorton AM-5610 [50] gives a figure showing the effect of illuminance on its open-circuit voltage. We simulate Formulas 3.17 and 3.18 to see how close the model is to the datasheet. The results are given in Figure 3.5.

We see that there is a difference between the modelled V_{oc} and the values given in the datasheet. The reason for this is the diode ideality factor, A, in Equations 3.16 and 3.18. A is a modelled value in both the four-parameter model and five-parameter model. If A is wrong, then the calculated V_{oc} value and thus the illuminance will be wrong. There is a difference between A from the four-parameter model and A from the five-parameter model. However, assuming the open-circuit voltage follows a logarithmic function in relation to the illuminance, we can remove this error by calibration.

A block diagram of Design 3 can be seen in Figure 3.6.

3.4.1 Calibration

In Design 3, we calibrate the system. To calibrate, we should add a calibration parameter to Equation 3.17, resulting in Equation 3.19, in which C is the calibration parameter.

$$G = \frac{G_0}{C} e^{\frac{V_{ADC} - V_{oc0}}{n_s A V_{T0}}} \tag{3.19}$$

This calibration method requires only one measurement to be done, and assumes that $G = 0$ at $V_{oc} = 0$. We calibrate the system by inputting a digital reference value of the illuminance. By comparing that value to the ADC value, the system can adjust C to match them.

If the open-circuit voltage of the solar cell can indeed be described by a purely logarithmic function, ignoring temperature effects, then this system in theory can be calibrated perfectly.

However, if the shunting resistance, R_{sh}, has a significant effect, as in Equation 3.18, then perfect calibration is not possible, and an error will remain, even after calibration. If R_{sh} is large enough, this should not have a significant effect.

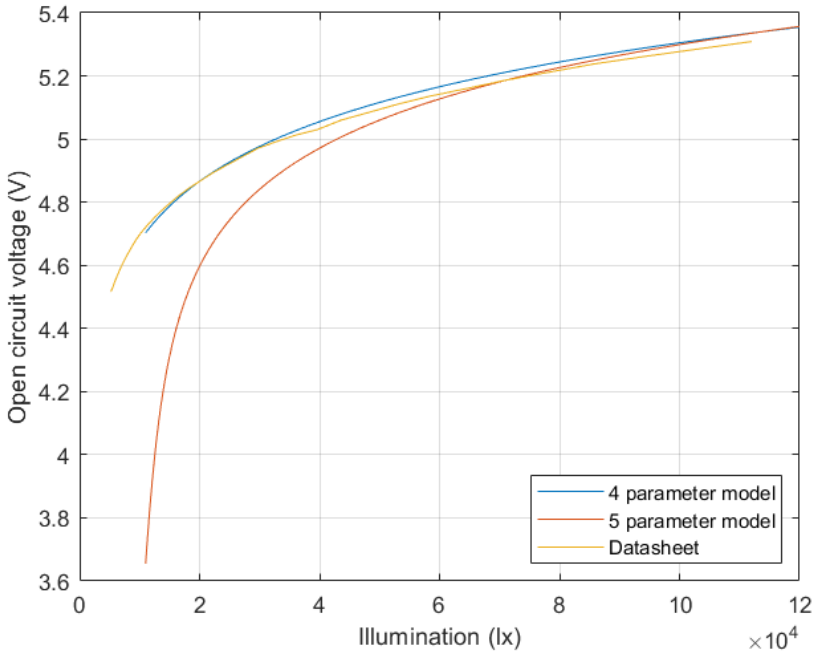

Figure 3.5: Open-circuit voltage comparison

3.4.2 Design limits

The open-circuit voltage design is limited by:

- Model accuracy
- Temperature
- The accuracy of the voltage-division circuit
- The accuracy of the ADC
- The accuracy of the calibration circuit

Model accuracy

The diode ideality factor depends on the model and is not completely accurate. However, calibrating the circuit should fix this error.

Temperature

Temperature has a pretty big impact on the open-circuit voltage, as the temperature coefficient, β, is relatively big. The voltage error is about $0.4\%/°\text{C}$, however, the dependency of temperature on illuminance is not linear, as given in Equation 3.8. For larger temperature variations the error gets much larger. However, for a range of about $10°\text{C}$ the estimate of $0.4\%/°\text{C}$ is reasonable. If the system is used only indoors, then this is a good assumption. For outdoor use, the error caused by temperature differences can get significant.

Voltage-division circuit

As mentioned, any of the named voltage-division techniques should suffice since all of them can be made more with a smaller error than 2.5%. Since the ADC is not more accurate than that, making this part of the system more accurate is unnecessary.

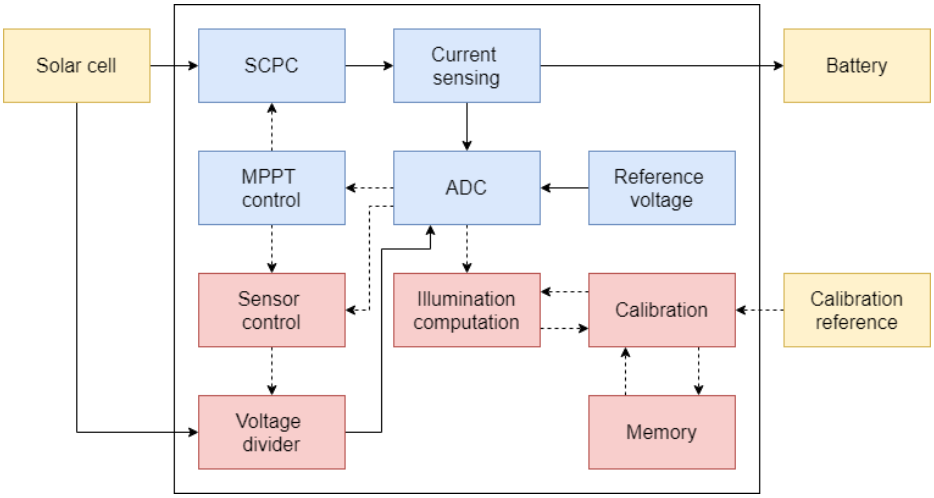

Figure 3.6: Design 3 block diagram

ADC accuracy

This method is limited by how accurately the ADC can detect the voltage. For a range of 0 to V_{oc}, this means you get an accuracy of $\frac{V_{oc}}{2^{ENOB}}$. Since the voltage is divided, it is possible to make multiple ranges in the voltage divider. This increases the possible accuracy of the system without the need for a better ADC. It is even possible to add an amplifier to increase the accuracy even more until noise starts to become dominant. This does not seem necessary, and instead, the smallest range of the system would be a divide by 1, for signals up to 0.8 V. For this range, the responsivity of the system when using the Panasonic Amorton-5610 would be 0.53 lx/V. This range does, however, fall outside of the range mentioned in the datasheet [50]. The lowest point in the datasheet is 4.5V, which corresponds to 5 klx. In this range, the responsivity is 1.1 klx/V, or 1.67 lx/LSEB. For solar cells designed for indoor use, the datasheet mentions a lower illuminance range. For the Panasonic Amorton-1456 for example, the responsivity at $V_{oc} = 2.21$ V, corresponds to 52.6 lx and results in a responsivity of 0.82 lx/LSEB [65].

Calibration

In Design 3, it is necessary to calibrate the system. We use the ADC in the calibration circuit, and thus the calibration accuracy is limited by the ADC.

3.4.3 Accuracy simulation

The inaccuracy of the ADC is about 2.5%. To see what error to expect for Design 3, we determine V_{oc} for the different points of illuminance. We add an error of 2.5% to V_{oc}, and then we simulate the error made. The results for the indoor and outdoor solar cells can be seen in Tables 3.7 and 3.8 respectively. The entries marked with an asterisk (*) fall outside the working range mentioned in the datasheet.

Table 3.7: Panasonic Amorton AM-1456 simulated error for Design 3

Range type	Illuminance (lx)	V_{oc} (V)	Error (2.5%)
Living room	50	2.22	28 lx
Office space	400	2.49	237 lx
Cloudy	1000	2.61	626 lx
Indirect sun	10,000	2.91*	6892 lx
Direct sun	100,000	3.22*	89,750 lx

Table 3.8: Panasonic Amorton AM-5610 simulated error for Design 3

Range type	Illuminance (lx)	V_{oc} (V)	Error (2.5%)
Living room	50	3.46	19 lx
Office space	400	4.02	170 lx
Cloudy	1000	4.27	460 lx
Indirect sun	10,000	4.91	6200 lx
Direct sun	100,000	5.53	66,790 lx

3.4.4 Conclusion

The advantage of Design 3 is that we only need a voltage measurement and that we can obtain the measurement during the normal functioning of the MPPT. Thus, it has little impact on the MPPT. The disadvantage is, however, that the error made in Design 3 is just under 50% at lower illuminance, and slightly above 50% for higher illuminance, as can be seen in Tables 3.7 and 3.8. This is accurate enough to give a label to it, like ”outdoor cloudy”, but not accurate enough to couple the result to an exact illuminance value. This design is therefore not usable for this project.

3.5 Design 4: MPP based design, measuring only current

Design 4 is similar to Design 1 and shares its advantages while getting rid of some of its disadvantages. The idea is that only a current measurement is done at the MPP. With that information, an estimation is made of the short-circuit current, which then can be used to calculate the illuminance according to Equation 3.15. Thus, it looks like a hybrid of Designs 1 and 2. Compared to Design 1, the advantages are that Design 4 does not need to measure the voltage, which also gets rid of the exponential calculation. Compared to Design 2, it has the advantage that the system does not need to stop the converter to do a measurement, and thus is more energy-efficient, while also having better integration with the MPPT. The disadvantage is that Design 4 is less precise than Design 2, as there will likely be an error when calculating the short-circuit current from the MPP current. Also, Design 4 assumes the system works at MPP. If the system does not then this will cause an error.

The estimation from I_{mpp} to I_{sc} is based on that the MPP current is always around 0.7 to 0.9 times the short-circuit current. In fact, this method is used for simple MPPT algorithms, as described in [48]. The short-circuit current is measured, which then gives information about the MPP current, according to Equation 3.20, in which D is a constant between 0.7 and 0.9.

$$I_{mpp} = DI_{sc} \tag{3.20}$$

In Design 4, we reverse this method. We measure the MPP current which then, via the same relation, gives information about the short-circuit current. C depends mostly on the specific solar cell, but also environmental conditions and degradation over time play a role. Calibrating with the specific solar cell, which is done anyway, should give a close result.

Since we already measure the SCPC output current in the NOWI NH16, it makes sense to use that measurement for the short-circuit current estimation. The SCPC has a ratio between the input and output current, so when using the output current measurement, we should compensate for this by dividing the measured current by the current conversion ratio.

A block diagram of Design 4 can be seen in Figure 3.7.

3.5.1 Design limits

The accuracy of Design 4 is limited by:

- Model accuracy
- Temperature
- The accuracy of the conversion ratio and losses in the SCPC
- The accuracy of the current-sensing circuit

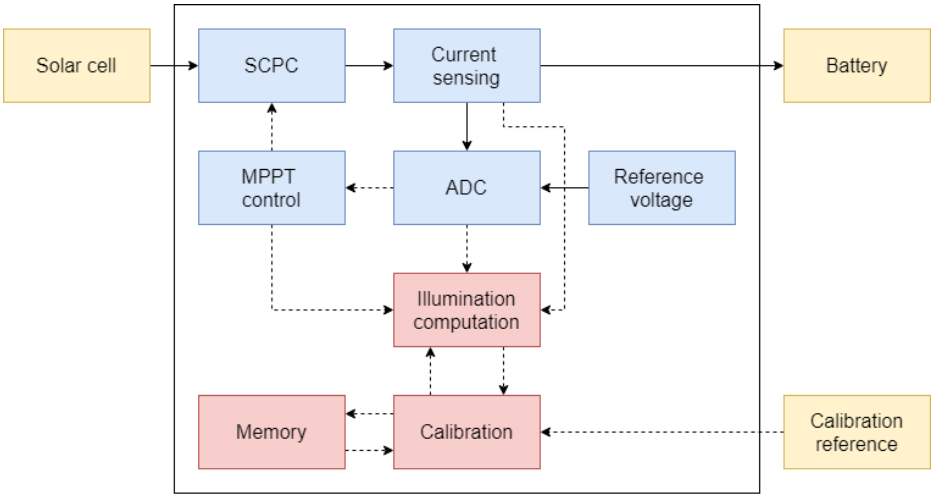

Figure 3.7: Design 4 block diagram

- MPPT precision
- The accuracy of the ADC
- The accuracy of the calibration circuit

Model accuracy

The model assumes a constant ratio between I_{sc} and I_{mpp}. This ratio is mostly constant but can change a bit under different temperature or illuminance conditions.

Looking at the datasheets for many different energy harvester solar cells, we calculate the expected ratio at different illuminance levels. Then we compare them to see what error to expect. The results can be seen in Table 3.9.

Table 3.9: Solar cell short-circuit current to MPP current ratio

Solar cell	Illuminance (lx)	I_{mpp} (A)	I_{sc} (A)	Ratio
Panasonic Amorton	114 k	4.7 m	5.5 m	0.86
AM-5610 [50]	50 k	2.1 m	2.4 m	0.89
Panasonic Amorton	200	7.1 μ	8.8 μ	0.81
AM-1454 [72]	50	2.7 μ	3.5 μ	0.79
Panasonic Amorton	200	4.8 μ	6.0 μ	0.79
AM-1456 [65]	50	1.2 μ	1.5 μ	0.78
	114 k	2.77 μ	342 μ	0.81
PowerFilm [73]	91 k	217 μ	266 μ	0.82
LL200-2.4-37	68 k	166 μ	205 μ	0.81
	46 k	107 μ	135 μ	0.79

The ratio is always around 0.8. It can vary between solar cells, but calibrating the system reduces that error.Remaining after calibration, is the error between different illuminance levels of the same solar cell. Looking at Table 3.9, we see that that error is up to 4%. This falls within the margin of what is acceptable. If the error would not be acceptable, calibrating at different illuminance levels could be a solution.

Temperature

The system is affected by the short-circuit current temperature coefficient, which is about 0.1%/°C for the Panasonic Amorton series. That means that over a temperature range of 0 up to 40°C, it would cause an error of up to 2% to either side.

Converter ratio and losses

The SCPC should convert the current input to output with a certain known ratio. However, due to its efficiency and gate charge losses this ratio is not exactly what is expected resulting in an error in the estimated short-circuit current. The SCPC accuracy is about 90-95%, which would result in an error of 5-10%. Calibration can partly solve this problem. There will still be an error left after calibration, however, and it is hard to say what this error will be exactly, but it is expected to be around 5%, as that is the range the SCPC efficiency varies.

Current-sensing circuit

The same applies as in Design 2. The current-sensing circuit should be accurate enough, but in the end, it is likely the ADC that is the true limit to the system's accuracy.

MPPT precision

Design 4 assumes that the MPPT is perfect and thus that the operating point of the solar cell is always at the MPP. In theory, if the MPPT has a constant error, this still should result in the same accuracy. However, if the MPPT varies around the MPP, which in practice it does, then this will result in an error in C of Equation 3.20. In the NOWI NH16, the current error caused by this stays within 2.5%. Since the system searches for the MPPT at discrete moments, the best accuracy can be achieved if the system uses current measurements from a moment directly after the MPP has been found. Otherwise if the incident light changes, this results in an error.

ADC accuracy

Since the ADC in this design also measures current, like in Design 2, the same errors as in Design 2 apply here. The responsivity is different, however, since in Design 4 I_{sc} is estimated using Equation 3.21.

$$I_{sc} = \frac{1}{C} I_{mpp} \tag{3.21}$$

The resulting responsivity is 25 klx$/$mA for the Panasonic Amorton AM-5610 and 33 klx$/$mA for the Panasonic Amorton AM-1456.

Calibration

We calibrate the system, and like in the other designs the ADC is likely the limit of the calibration accuracy. The calibration cycle needs to take place directly after an MPP has been found, otherwise if illuminance changes in between the MPPT cycle and calibration cycle, this will result in an error in the calibration.

3.5.2 Calculation

The parts of the circuit influencing the illuminance calculation in Design 4 are:

- Solar cell responsivity
- Charge-pump current conversion factor
- Effective resistance of the current sensor
- ADC bit to voltage ratio

Solar cell responsivity

The solar cell responsivity is the amount of measured current that corresponds to the illuminance of the solar cell. This number links the illuminance to the solar cell MPP current, or $G = G_0 \times D \times \frac{I_{mpp}}{I_{sc0}}$, in which D is the ratio between I_{mpp} and I_{sc}. D depends on the specific solar cell and will be calibrated for.

Charge-pump current conversion factor

The charge pump boosts the voltage, however, this means the current is attenuated. The multiplication factor with which this happens is known in the system and thus can be corrected for. $I_{mpp} = B \times I_{cp,out}$, in which B is the charge-pump voltage boost factor. The value of B is known as it is set beforehand. There are some small current losses in the charge pump, meaning B is not exactly what is expected, however, calibration helps against this.

Effective resistance of the current sensor

The output current of the charge pump is translated to a voltage by the current-sensing circuit, as the ADC works with voltages. This is done via a senseFET circuit, which copies (part of) the current, and uses a sense resistor to create a proportional voltage. The value of the current mirror ratio and sense resistor can be changed by the system to facilitate a larger current range. The system has five different ranges with a different current mirror factor and sense resistor. This means the system has four different effective resistances, being 100 kΩ, 10 kΩ, $736\ \Omega$, and $48\ \Omega$. The relative ratio between the effective resistances should be within the 2.5% inaccuracy of the NOWI NH16. Calibration is used to remove the absolute error. If necessary, it is possible to calibrate for every range, however, as the relative accuracy is good enough, this should not be necessary.

ADC bit to voltage ratio

The ADC gives an output between 0-255. This output corresponds to a voltage between 0-0.8 V. We do the calculation with the output bits of the ADC, which correspond to a multiplication factor of $\frac{0.8}{255} = 3.14$ mV/bit.

The total calculation is done according to Equation 3.22,

$$G = \text{ADC}_{bits} \times 0.00314 \times \frac{1}{R_{eff}} \times \frac{1}{b} \times \frac{G_0 \times C}{I_{sc0}} \tag{3.22}$$

in which C is the calibration constant calibrating for the solar cell responsivity, together with any other gain errors in the charge pump and current sensor. This also includes the ratio between I_{mpp} and I_{sc}, D.

3.5.3 Accuracy simulation

The error made in Design 4 in measuring I_{mpp}, is the same as for the short-circuit design when measuring I_{sc}. Since I_{mpp} is used to estimate I_{sc}, and I_{sc} is used to estimate the illuminance, with the same gain error, the same error in lux will be made. However, this system also introduces a possible model error of up to 4%. Combined with the possible measurement error, this would amount to a maximum error of $\sqrt{2.5^2 + 4^2} = 5\%$. We simulate the error this creates and the result can be seen in Tables 3.10 and 3.11 respectively.

Table 3.10: Panasonic Amorton AM-1456 simulated error for Design 4

Range type	Illuminance (lx)	I_{mpp} (μA)	Error (2.5%)	Error (5%)	Error (10%)
Living room	50	1.5	1 lx	3 lx	4 lx
Office space	400	12	10 lx	20 lx	40 lx
Cloudy	1000	30	25 lx	50 lx	100 lx
Indirect sun	10,000	300*	250 lx	500 lx	1000 lx
Direct sun	100,000	3000*	2500 lx	5000 lx	10,000 lx

Table 3.11: Panasonic Amorton AM-5610 simulated error for Design 4

Range type	Illuminance (lx)	I_{mpp} (mA)	Error (2.5%)	Error (5%)	Error (10%)
Living room	50	0.002	1 lx	3 lx	4 lx
Office space	400	0.016	10 lx	20 lx	40 lx
Cloudy	1000	0.041	25 lx	50 lx	100 lx
Indirect sun	10,000	0.41	250 lx	500 lx	1000 lx
Direct sun	100,000	4.1	2500 lx	5000 lx	10,000 lx

Since the model is linear, logically the simulated error is the same for both. Values marked with an asterisk (*) fall outside of the working range according to the datasheet.

3.5.4 Conclusion

The accuracy of Design 4 is not as good as Design 2, however, it is better than the accuracy of Designs 1 and 3. The advantage over Design 2 is that Design 4 has a lower impact on the MPPT, making it easier to integrate. Also, since the harvester can keep harvesting during its illuminance sensing cycle, Design 4 is more energy-efficient than Design 2.

3.6 Design comparison

A simple comparison of the four designs can be found in Table 3.12.

Table 3.12: Comparison of the four mathematical design options

Design	Accuracy	MPPT integration	Computational complexity	Energy efficiency
1	-	+	-	+
2	+	-	+	-
3	-	0	0	0
4	0	+	+	+

The fourth design has the best overall potential for this project. The accuracy is not the best, however, it is good enough. Added advantages are, that Design 4 can be integrated easily with the current MPPT system and is more energy-efficient than the other designs. Therefore we use Design 4.

4. Hardware design

Now that we have chosen a design option, we make it into hardware. Looking at the block diagram of the design in Figure 3.7, the parts we need to make are the illuminance computation and calibration blocks.

4.1 Design tools

The design includes a digital calculation and digital calibration, from the data outputted by an ADC. Thus, digital design tools will be used to create the hardware. We make the design in Verilog, which we simulate and verify using ModelSim. Then we synthesise the design using Cadence Genus. Next, to verify the synthesised design, we simulate it using Cadence Spectre. Also, we make a prototype of the design on an FPGA. The FPGA used is a Zybo Z7 and we program it using Xilinx Vivado.

4.2 Floating-point design

The illuminance can range from 1 lx to about 200 klx. Since the precision of the system is about 10%, there is no need to be able to represent 200,000 different numbers. Instead, two significant (decimal) numbers should be good enough. Therefore, we explore a floating-point design.

4.2.1 Floating-point multiplication

Some of the factors multiplied are below 1, and thus to get a workable number, floating-point multiplication has to be done. This also has the advantage that for bigger numbers, near 200,000, fewer bits are needed. In digital systems a floating-point number has three parts:

- A sign bit, S
- An exponent, E
- A fraction, called the mantissa, M

The floating-point number then is $(-1)^S \times M \times 2^E$, which is positive if $S = 0$, and negative if $S = 1$. In our system, we only work with positive values, so the sign bit is not necessary.
The mantissa is normally 23 bits, however, the system is not precise enough to need this amount of bits. Instead, we choose a number such that 3 significant (decimal) numbers can be guaranteed in the calculation, which requires 10 bits, as $2^{10} = 1024$.
Then we choose the exponent such that the entire expected illuminance range up to $200,000$ lx can be reached. To be able to handle numbers up to 200,000, we need 5 exponent bits, as $2^{2^5} > 200,000$. The initial exponent can, however, be negative, and to also be able to represent these numbers, we need an extra bit. Therefore the exponent consists of a 6 bit two's complement number.

A more optimised design is possible, needing only two significant (decimal) numbers, and using the exponential part more efficiently. So the number of bits used could be reduced, however, for a first design, 11 bits should suffice.

The floating-point design of the illuminance calculation hardware consists of the following elements:

- Look-up table (LUT)
- Adder
- Multiplication block including controlling FSM
- Calibration FSM

A block diagram of the floating-point design can be found in Figure 4.1.

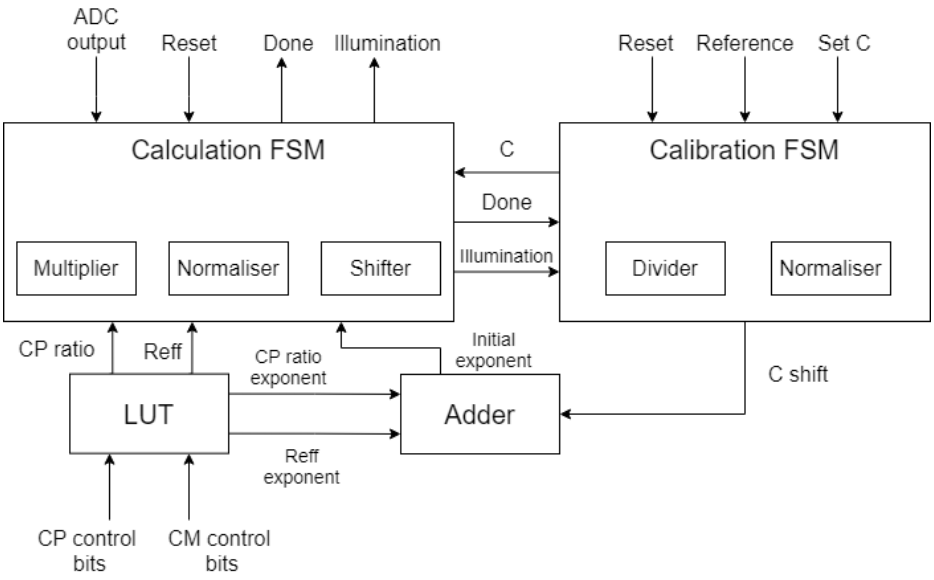

Figure 4.1: Floating-point design block diagram

4.2.2 LUT

As stated before, the current in the NOWI NH16 will first go through the charge pump and then through a current mirror with a sense resistor. The charge-pump conversion ratio lowers the current by more or less that ratio. The current mirror, together with the sense resistor, translate this current to a corresponding voltage. The ratio between the current going into the current mirror and the voltage is determined by the division factor of the current mirror, and the value of the sense resistance. Together, this ratio can be seen as the effective resistance of the current mirror, R_{eff}, in which $V_{out} = I_{in} \times R_{eff}$. In the NOWI NH16 four different values for R_{eff} exist.

Since the charge-pump ratio and the current-mirror effective resistance have a limited number of discrete options they can be put into a LUT. Based on the controlling bits of the NOWI chip, the LUT will output the correct values for the charge-pump ratio and R_{eff}. Another advantage is that the reciprocal of the effective resistance, or effective conductance, G_{eff}, can be put into the LUT instead of R_{eff}. This means that we do not need a division, but instead, we can use a multiplication, which is smaller and is easier to implement.

The charge-pump conversion ratio values are: 2x, 4x, 8x, and 16x. They are represented by a 2-bit control signal, ”CP control bits” in Figure 4.1, in which 2x is represented by ”00”, 4x by ”01”, 8x by ”10” and 16x by ”11”. Converting the ”CP control bits” to a logarithmic number is simple, as $2^1 = 2$, $2^2 = 4$, etc. Therefore the number can be converter to a logarithm by adding 1 to the control bits. All values at the decimal places are zero.

4.2.3 Adder

The adder adds the exponential part from the LUT entries and the calibration parameter, to form one number. This number forms an initial exponential part for the illuminance calculation.

4.2.4 Illuminance calculation

The calculation of the illuminance can be found in Equation 3.22. It consists of five parts of which the ADC bits to voltage factor, 0.00314, and $\frac{G_0 \times c}{I_{sc0}}$ are constant. These can be combined into a single number. The calculation then involves the following steps:

- Getting the ADC bits
- Getting the reciprocal of the effective resistance from the LUT
- Getting the charge-pump voltage boost factor from the LUT
- Multiplying all these values together with the constant

The calculation hardware consists of an FSM with the following components:

- Multiplier
- Normaliser
- Shifter

Multiplier

The multiplier is the element that actually does the multiplication. This multiplier has two 10-bit inputs, as it is used to multiply two mantissae together. The output of the multiplier is 20 bits so that every possible multiplication fits. Different ways of multiplication in digital systems exist, usually being a trade-off between area and latency. In this project, a smaller area is more important. The following multiplier implementations can be used:

Basic operation The most basic multiplication method of $A \times B$ is by checking the LSB of B, and adding A to the final result if this is a 1. Then it left-shifts A and checks the LSB-1 from B, and so on. This method is not the fastest but can be efficiently implemented by synthesisers.

Booth's algorithm Booth's algorithm makes use of the fact that an adder can also subtract by adding the inverse in two's complement. When multiplying by $2^x + 2^{x-1} + ... + 2^{x-n}$, this is equal to multiplying by $2^{x+1} - 2^{x-n}$. The latter, however, requires fewer operations. In this method the algorithm can work with two bits at the same time, in other words, it will add the multiplicand if the last two bits are 01, and subtract the multiplicand if the last two bits are 10, and do nothing otherwise, right-shifting B after every step.

Wallace tree A Wallace tree multiplies by multiplying every bit of A with every bit of B. Then adders and half adders are used to reduce the number of partial products, by adding the n^{th}-bit of two (half-adder) or three (full-adder) partial products, reducing the number of partial products until any bit position only has up to two bits left. This then leaves two numbers, which are then added with a normal adder.

The advantage of this system is that it is fast. The disadvantage is that a lot of adders are required, which requires a large area.

Dadda multiplier The Dadda multiplier works almost the same as the Wallace tree. The difference is that the Dadda multiplier optimises the hardware needed by using the minimal number of adders in the reduction stage. This is done by reducing the number of layers after calculating the partial products. For a Wallace tree, the number of layers is equal to the number of partial products. However, the Dadda multiplier reduces the number of layers by moving bits from the bottom layers to empty spots on higher layers.

Conclusion Since speed is not the main focus of this project, we will use the basic multiplication.

Normaliser

The output of the multiplier is a 20-bit number, however, the multiplier input can only handle 10-bit numbers. Thus, we should normalise the number to 10 bits. We do this by comparing the top 10 bits and if there is a one in any of them, we right-shift the entire number and increase the shift register. The shift register keeps track of the exponent. The right-shifting continues until no bit is 1 above the 10^{th}-bit position anymore. If no bit is 1 in the top 10 bits and the 10^{th}-bit is 0, then the number is left-shifted and the shift register is decreased. This continues until the 10^{th}-bit position is a 1.

On reset, we initialise the shift register to the number the adder has calculated.

Shifter

After all numbers are multiplied with each other, we convert the floating-point number to an unsigned binary number used to describe the illuminance. This conversion is done by the shifter. Essentially the shifter does the opposite of the normaliser. If the shift register is negative then the number is right-shifted, and one is added to the shift register until its value is 0. If the shift register has a positive value, the number is left-shifted, and the shift registers value is decreased by one until the shift registers value is at 0.

FSM

The FSM ties together all components above. A flowchart of the functioning of the FSM can be found in Figure 4.2. The system multiplies two numbers together, then enables the normaliser to normalise the number. The system multiplies this number again and normalises it again. The system then multiplies the number and forwards the answer to the shifter, which generates the final result.

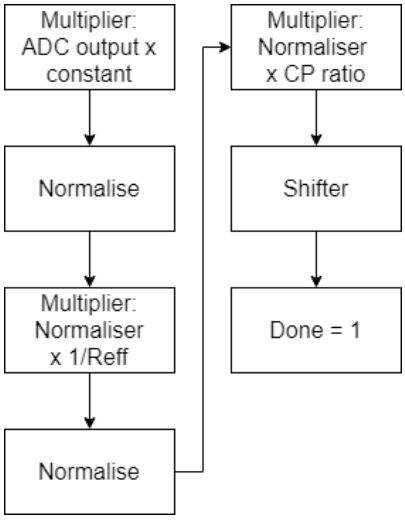

Figure 4.2: Multiplier FSM flow chart

4.2.5 Calibration

We calibrate the system. To do so, a certain amount of light must shine on the solar cell. We measure this light externally, and we input the result, in lux, into the system as a reference. The system compares the reference to the calculated illuminance, and if the values are not the same, it alters the calibration parameter. Looking at Equation 3.22, parameter C can be determined by dividing the reference by the result of the calculation when $C = 1$. Therefore we need a division circuit. The output of this block is an input for the illuminance calculation.

The calibration block consists of an FSM with the following elements:

- Division circuit
- Normaliser

Division

To calibrate the system, we use a divider. Several division algorithms exist for digital systems. In [74], an overview of possible algorithms is given. The algorithms described are:

- Restoring and non-restoring algorithms
- SRT division
- Taylor's series expansion
- Goldschmidt's algorithm
- Newton-Raphson algorithm
- CORDIC algorithm

Restoring and non-restoring algorithms These are relatively simple algorithms based on $N = MQ+R$, with N the dividend, M the divisor, Q the quotient, and R the remainder. These algorithms subtract M until the remainder is smaller than the dividend. The difference between the two is that the restoring algorithm does a test subtraction each cycle, and has to do restoration if the remainder drops below zero, while the non-restoring algorithm does not, but needs an extra bit to keep track of the sign of the remainder.

In practice, the non-restoring algorithm requires fewer cycles than the restoring algorithm.

The advantage of these algorithms is that they are relatively simple, and thus require few resources. Only an adder and some comparison logic are required. The disadvantage of these algorithms is that they require more cycles than some of the more advanced algorithms, especially for more precise numbers.

SRT division SRT division uses a LUT dependent on the divisor and dividend, usually with a redundant amount of entries. This way each iteration a decision can be made based only on an amount of the most significant bits. Because of the redundancy, any error can be corrected in later cycles, which speeds up the division process but does require a LUT.

Taylor's series expansion This method is based on a Taylor's series expansion, as $Q = \frac{N}{M} \approx NX_0(1 + (1 - MX_0) + (1 - MX_0)^2 + (1 - MX_0)^3)$. Adding more terms will make a more precise result, but also will take longer or require more hardware. Taylor's series expansion requires the use of squaring and/or cubing units to make the algorithm fast, and thus requires a larger area. This method is not often used [75].

Goldschmidt's algorithm Goldschmidt's algorithm uses parallel multiplication of both the dividend and the divisor, over multiple iterations until the divisor converges to one. At that point, the dividend gives the quotient. The disadvantage of this method is that it requires the use of two multipliers, meaning a large area is needed. This method is, however, fast, and used in many microprocessors.

Newton-Raphson algorithm The Newton-Raphson algorithm finds the reciprocal of M and multiplies that with N to find Q. To do so, an estimation of the reciprocal is made, and in following iterations, more precise estimations are made until the final result is found. Needed for this method is a function that, over several iterations, converges to the reciprocal of the dividend. Used is $X_{i+1} = X_i - \frac{f(X_i)}{f'(X_i)}$, with $f(X_i) = \frac{1}{X} - M$, in which X needs to converge to $\frac{1}{D}$, so $f(X)$ needs to be zero at $X = \frac{1}{M}$.

The hardware required for this method is significant since a multiplier unit and a subtraction unit are needed. Multiple multipliers can be used to make a faster implementation. This method has a bigger advantage for calculating larger numbers, as the number of correct digits doubles every iteration. However, for smaller numbers, this method can be relatively slow.

CORDIC algorithm The CORDIC algorithm can be used to implement many trigonometric functions, including division, however, if only a division is needed the CORDIC algorithm is not area or power efficient.

Conclusion Since this project focuses on the smallest area, we use the non-restoring algorithm to implement the divider.

Normaliser

We use the normaliser to normalise the reference input. It is possible to combine this normaliser with the normaliser in the illuminance computation block with the right controlling hardware. Another option is that the user inputs the reference in floating-point format, which would mean we do not require this normaliser at all.

Calibration FSM

The calibration FSM first normalises the reference and the illuminance calculation. It is possible to skip that last step by directly connecting the normalised output of the multiplier FSM to the calibration FSM. However, this requires extra wires, and the extra time gained is not critical for the application.

The calibration FSM then checks if the dividend is larger than the divisor, and if it is, the calibration FSM left-shifts the dividend one bit. Now the dividend is between 1 and 0.5, while the divisor is between 1 and 2. The calibration FSM then sends these numbers to the divider, and subtracts the exponent of the divisor from the exponent of the dividend, to get the final values for the calibration parameter C.

4.2.6 Synthesis

We synthesised the design using Cadence Genus. The logic cell area can be found in Table 4.1. The area given is the cell area only, and making the layout of the system will increase the total area. The area given for the Top-level is the total area.

Table 4.1: Synthesis of the floating-point design of the illuminance calculation hardware

Module	Submodule	Cell count	Cell area (μm^2)
Top-level	-	1625	77321
	Calculation FSM	688	34016
	Calibration FSM	877	40972
	LUT	27	1433
	Adder	33	900
Calculation FSM	Shifter	127	5849
	Normaliser	188	8072
	Multiplier	198	9894
Calibration FSM	Divider	374	16150
	Normaliser	183	7878

4.3 Logarithmic design

The first design was based on floating-point numbers to get accurate numbers over a range from 1 to 200,000. Since, for bigger numbers, only the first two numbers are significant, as the system has an error of about 10%, using a logarithmic scale makes more sense. This design reduces the number of bits, which makes the entire system smaller, especially the multiplication circuit. And since in the logarithmic domain multiplications and divisions are additions and subtractions, the divider can be omitted completely. Instead, we need a binary to base-2 logarithm converter, and possibly a base-2 logarithm to binary converter. Alternatively, we can represent the illuminance result by a logarithmic number.

To represent numbers up to 200,000, we need five bits for representing integer numbers, as $2^4 = 16$ and $2^5 = 32$, while $2^{18} > 200,000$. And $16 < 18 < 32$. It is possible to reduce this amount by one bit, by shifting up the entire range eight entries, so by omitting the eight lowest values. This is not unreasonable, as the lowest values are beyond the working range of the solar cell and thus would not give an accurate result anyway. By shifting the entire range, the number $2^{16+shift} > 200,000$, while at the lower range numbers with a value of one, two, or three

are discarded.

For example using 6 bits, four above 1, two below, meaning you use the numbers 0, 0.25, 0.5, ..., 15.75. In this the numbers 0, 0.25, and 0.5 all result in the number 1 when rounded: $2^0 = 1$, $2^{0.25} = 1.2$, $2^0 = 1.4$. The highest number now is $2^{15.75} = 55,109$. Instead the highest number needed is 17.75, since $2^{17.75} = 220,436$. Instead of adding another bit, we could also shift all the numbers by eight places, meaning essentially 2 is added to each number. In this case, the lowest number becomes $2^2 = 4$, and the highest number now is $220,436$. This is justifiable since being able to differentiate between 1 and 4 lux is not very important.

To get two significant numbers over the entire range, six non-integer numbers are needed, to have a logarithmic number for every combination of two significant numbers.

The hardware of the system consists of the following elements:

- LUT
- Binary to base-2 log converter
- Calculation FSM

A block diagram of the LUT-based design can be found in Figure 4.3.

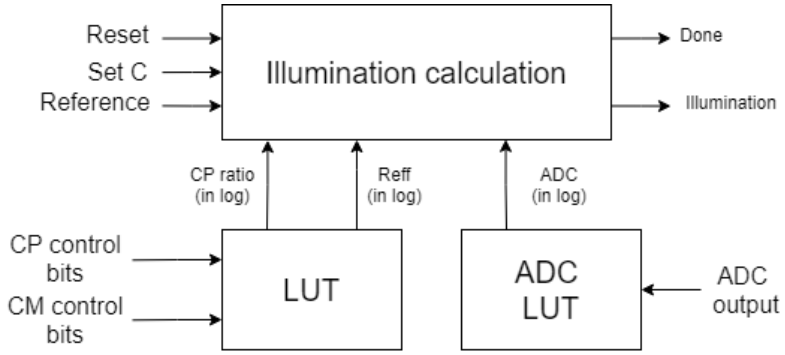

Figure 4.3: LUT-based design block diagram

4.3.1 LUT

Like in the floating-point design, we use the LUT to find the correct values of the charge-pump conversion ratio and R_{eff}. In this case, we store R_{eff} and not the reciprocal, as storing the reciprocal on a log scale would require a negative number. This way we do not need negative numbers in the system, saving a bit.

4.3.2 Binary to base-2 log converter

We need a binary to base-2 log converter to convert the ADC output to base-2 logarithm. One way to do this is via a conversion algorithm. Another option is via a LUT-based conversion.

Algorithm-based conversion

We have designed the binary to base-2 log converter as described in [76]. The algorithm requires a squaring operation. At first, a multiplier was used to implement this operation, which was similar to the multiplier of the previous design, but with fewer bits. However, as bits reduce, purely combinatorial logic starts to get more efficient. Synthesis shows that a fully combinatorial multiplier is more efficient than the pipelined multiplier approach used in the previous design. On top of that, Cadence Genus has a special block used to synthesise squaring units, decreasing the area even more.

The operation works as follows, see Figure 4.4:

- The logarithm's integer part is set to six.

- If the highest bit of the ADC output is one, then the output is right-shifted by one position, and one is added to the logarithm's integer part.
- Else, if the 7th-bit of the ADC output is not a one, the output is left-shifted, and one is subtracted from the logarithm's integer part until the 7th-bit is one. The number is now normalised between 1 and 2.
- The normalised ADC output, m, is squared, and the final result, z, is multiplied by 2, which is implemented with a left-shift.
- Now, if m is 2 or more, z is increased by 1, and m is divided by 2, which is implemented with a right-shift.
- The previous two steps are repeated for a set number of iterations, equal to the number of bits, so in this case 6 times.

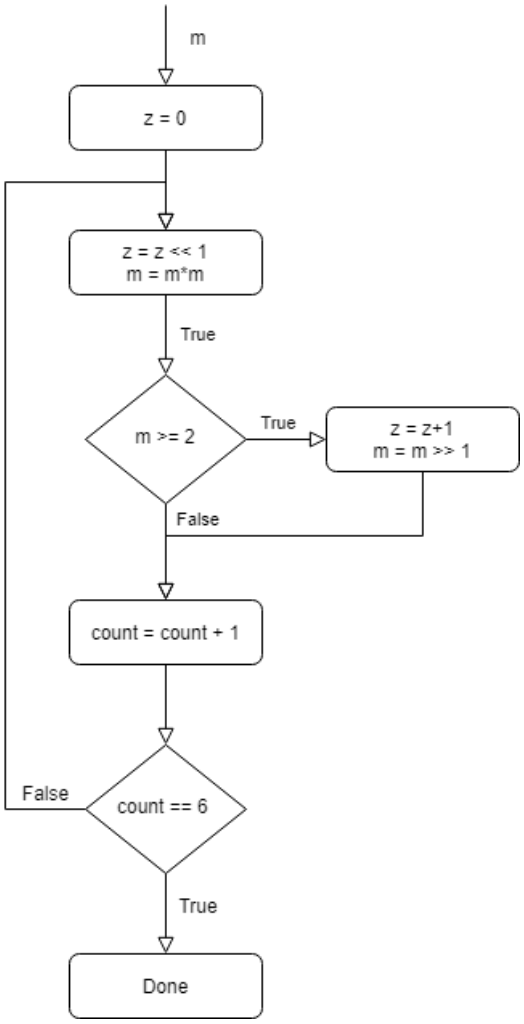

Figure 4.4: Binary to base-2 logarithm converter

The advantage of the algorithm-based method is that the area of the circuit does not get much bigger if we add more bits. A disadvantage is that right-shifting truncates the numbers, and thus, we lose some accuracy. In fact, when calculating the error it turned out the average error is less if at the end of the conversion one LSB is added, resulting in an average error of 0.5 as opposed to 1.43, over the entire range from 0 to 255.

LUT-based conversion

Another way of converting the ADC output to base-2 log is via a LUT containing all possible ADC outputs and their corresponding logarithmic number. The ADC has 256 different possible outputs, all of which can be converted

to a base-2 logarithmic number. Synthesis of this LUT shows that with a logic cell area of $6780\ \mu m^2$, this circuit is smaller than the algorithm-based conversion method. Another advantage is that this method is faster than the algorithm-based method, as it executes in a single cycle. The LUT-based converter is not clocked.

4.3.3 Calculation FSM

The FSM in this design takes care of both calculating the illuminance and doing the calibration. The FSM takes the following steps:

- If the system is not yet calibrated, the system will look if the option to set the calibration parameter is enabled. If it is, the reference input is set as the calibration parameter.
- Then the binary to base-2 log converter is enabled, converting the ADC output to base-2 logarithm. For the LUT-based design, this step is skipped, as the LUT already converted the ADC output to a base-2 logarithm.
- The illuminance is then calculated by adding C, the converted ADC output and the charge-pump ratio, and then subtracting R_{eff}.
- If the system was not yet calibrated, the result is subtracted from the reference and the initial value for C is added. This is then set as the new value for C. The FSM will now reset and calculate the illuminance again.
- Otherwise the number calculated is the final result for the calculated illuminance. The system will signal that the calculation is done, and reset for the next calculation.

On the chip, the reference can be inputted via I2C. The system then reads this data from the register and use it for calibration. Each time a new reference is received, the system will reset its calibration flag, and do the calibration again. The reference should be inputted in base-2 logarithm.

4.4 Synthesis

The design has been synthesised using Cadence Genus. Cadence Genus reports on area, power, and timing. The file to initialise synthesis can be found in Appendix H. With that file, the Verilog files from Appendices B and C are synthesised.

4.4.1 Area

The cell area for the design with the algorithm-based converter can be found in Table 4.2, and the cell area for the design with the LUT-based converter can be found in Table 4.3. Again, the area given is the cell area only, and the area given for the Top-level is the total area.

Table 4.2: Synthesis of the logarithmic design with algorithm-based converter

Module	**Sub module**	**Cell count**	**Cell area (μm^2)**
Top-level	-	427	16866
	Calculation FSM	415	16599
	LUT	12	267
Calculation FSM	To base-2 log converter	232	8953
To base-2 log converter	Squaring unit	63	1904

We see that the LUT-based design has a smaller total logic cell area than the algorithm-based design. The difference is $2286\ \mu m^2$.

Table 4.3: Synthesis of the logarithmic design with LUT-based converter

Module	Submodule	Cell count	Cell area (μm^2)
Top-level	-	463	14580
	Calculation FSM	188	7533
	LUT	12	267
	ADC LUT	263	6780

4.4.2 Power

The power usage as estimated by Cadence Genus can be found in Tables 4.4 and 4.5, for the algorithm-based converter and LUT-based converter respectively. The power is estimated for a 1 kHz clock.

Table 4.4: Power report of the logarithmic design with algorithm-based converter

Module	Sub module	Static power (nW)	Dynamic power (μW)
Top-level	-	171	1294
	Calculation FSM	169	1160
	LUT	2	60
Calculation FSM	To base-2 log converter	88	109
To base-2 log converter	Squaring unit	20	36

Table 4.5: Power report of the logarithmic design with LUT-based converter

Module	Sub module	Static power (nW)	Dynamic power (μW)
Top-level	-	117	4083
	Calculation FSM	79	2901
	LUT	2	68
	ADC LUT	36	980

According to Cadence Genus, the power used by the LUT-based converter is more than twice the power of the algorithm-based converter. However, the algorithm-based converter needs 34 clock cycles per conversion, while the LUT-based converter needs only three. This means the energy per conversion of the LUT-based system is about four times lower than the algorithm-based converter.

4.4.3 Timing

Cadence Genus has also done a timing analysis. However, the clock period is much longer than the critical path of the system, and the system is therefore not time-critical. The system works on a 1 kHz clock. This means the critical path has 1 ms to complete. According to Cadence Genus, the critical path takes about $8\ \mu$s and thus has a slack of $992\ \mu$s. Thus, timing is not critical to the application.

4.5 Conclusion

We have discussed two different implementations of the design chosen in Chapter 3, in which the logarithmic implementation has advantages in size and power over the floating-point design.

Then we made and synthesised two different implementations of the logarithmic design. Here, the design with LUT-based conversion proved advantageous over the algorithm-based design in area, power, and delay.

In the end, we synthesised the design, and it has an area of 1.46 mm^2.

5. Verification

The last step is to verify the design of the illuminance calculation hardware made in Chapter 4. We do this in several steps. First, we verify the design with event-based simulation in ModelSim. Then we verify the system via circuit simulation in Cadence. We also verify the system in combination with the charge pump, current sensor, and ADC of the NOWI NH16, in Cadence. Lastly, we verify the system on an FPGA.

The data used for these simulations is both simulated values, and measured data from two solar cells typically used in energy-harvesting applications.

5.1 Verification dataset

To verify the system, we do simulations with both a solar cell model as input, but also with solar cell data as input. For that last part, we need measurement data. Specifically, we need to know the MPP current of a solar cell at different illuminance values. For this purpose, we use a set of three Panasonic Amorton AM-1456 [65] cells in parallel, and a Panasonic Amorton AM-5308 [77] solar cell. Also, we use a Sekonic Spectromaster C-7000 to measure the illuminance, some resistors and a potentiometer to get the solar cells to their MPP, and two multimeters to measure the current and voltage.

The steps to measure the illuminance and corresponding MPP current are as follows:

1. Place the solar cell at the same height as the spectrometer on a horizontal surface
2. Connect the multimeters and the resistors
3. Change the resistance until the solar cell is at MPP
4. Log the current measured
5. Measure the illuminance with the spectrometer at the same position as the solar cell
6. Repeat this process at various illuminance levels

The measured data can be found in Tables 5.1 and 5.2, for the Panasonic Amorton AM-1456 and AM-5308 respectively.

Table 5.1: Panasonic Amorton AM-1456 solar cell measured MPP current vs illuminance

Illuminance (lx)	188	322	419	556	587	848	887	1080	1340	1660	1970
I_{mpp} **(μA)**	17.12	26.38	37.06	51.39	51.97	96.10	98.57	123.1	152.1	153.1	179.8
Illuminance (lx)	2460	3090	3550	4610	5070	6130	7740	9070	25200	27600	42600
I_{mpp} **(μA)**	217.3	275.7	297.6	370.6	401.9	554.7	562.8	662.4	1151	1309	1702

5.2 Event-based simulation

We simulate the Verilog design by using ModelSim SE 10.1b. The Verilog files for the LUT-based design can be found in Appendix B. The Verilog files for the algorithm-based design can be found in Appendix C. We have

Table 5.2: Panasonic Amorton AM-5308 solar cell measured MPP current vs illuminance

Illuminance (lx)	186	272	538	599	626	674	731	952	1030	1160
I_{mpp} **(**μ**A)**	88	141	318	388	354	417	448	588	652	750
Illuminance (lx)	1190	1390	2420	2460	3090	4370	4990	6030	6120	6860
I_{mpp} **(**μ**A)**	768	883	1414	1472	1949	2484	2855	3747	4015	4630
Illuminance (lx)	7810	8320	9360	9830	10600	12200	14600	19000	24500	25800
I_{mpp} **(**μ**A)**	4619	4845	6240	6460	6988	7847	8200	13000	13900	16400
Illuminance (lx)	27500	30000	33700	34400	43400					
I_{mpp} **(**μ**A)**	16880	17800	20100	20920	28300					

chosen a current value of $360\ \mu$A for this simulation. This value is in the middle of the middle range of the NOWI NH16, thus corresponding to an ADC value of 127. At this range $R_{eff} = 732\,\Omega$. Then, via MATLAB simulation of the Panasonic Amorton AM-5610, we find the correct value for the illuminance. This value is 9543 lx, which we use in the simulation for the reference value. 9543 corresponds to a base-2 logarithmic value of 13.22, represented in the system by the number 846, which is inputted as the reference value. We have set the charge pump conversion ratio at 2x.

We vary the ADC output, and we compare the simulation result to a calculated result in Excel to check if the system works correctly. The results can be found in Appendix E. The full Excel sheet used for the calculation can be found in Appendix F. Also, we have varied the possible values for the charge pump conversion ratio and R_{eff}. Comparing the output result to the excel file gives the exact same result, and thus we conclude that the system works perfectly in the ideal case, meaning the digital signals are perfect ones and zeros, and there is no clock skew or jitter.

Looking at Appendix E, this means that when an ADC value from Column 12 'ADC output' is given, the resulting number calculated by the illuminance calculation hardware is the same as the number from Column 11 'Out full'.

We run the same simulation with the measured solar cell data from Tables 5.1 and 5.2. In this simulation, we use the same calculation as before to calculate the ADC value for each measurement. We simulate the illuminance calculation hardware with these calculated ADC values and calculate the resulting illuminance from the simulated output. The results can be found in Tables 5.3 and 5.4.

Table 5.3: Event-based simulation with measured solar cell data from Table 5.1

Digital output	Calculated illuminance (lx)	Measured illuminance (lx)	Error (%)
489	200	188	6.1
526	298	322	7.5
559	426	419	1.6
589	589	556	6.0
590	596	587	1.5
623	852	848	0.4
622	843	887	5.0
650	1141	1080	5.7
663	1314	1340	2.0
692	1798	1660	8.3
704	2048	1970	4.0
720	2435	2460	1.0
743	3124	3090	1.1
751	3407	3550	4.0
770	4186	4610	9.2
778	4565	5070	10.0
808	6317	6130	3.0
810	6455	7740	16.6
825	7594	9070	16.3
876	13193	25200	47.6
887	14862	27600	46.2
912	19484	42600	54.3

Table 5.4: Event-based simulation with measured solar cell data from Table 5.2

Digital output	Calculated illuminance (lx)	Measured illuminance (lx)	Error (%)
458	143	186	23.3
501	227	272	16.5
575	506	538	5.9
593	616	599	2.8
585	564	626	9.8
601	671	674	0.4
607	716	731	2.0
633	949	952	0.3
629	909	983	7.5
642	1046	1030	1.6
655	1205	1160	3.8
657	1231	1190	3.4
670	1417	1390	2.0
714	2282	2420	5.7
717	2358	2460	4.2
743	3124	3090	1.1
765	3965	4370	9.3
778	4565	4990	8.5
804	6049	6030	0.3
810	6455	6120	5.5
824	7512	6860	9.5
824	7512	7810	3.8
826	7677	8320	7.7
850	9955	9360	6.4
854	10396	9830	5.8
861	11215	10600	5.8
873	12771	12200	4.7
876	13193	14600	9.6
918	20792	19000	9.4
924	22188	24500	9.4
940	26386	25800	2.3
942	26964	27500	1.9
948	28774	30000	4.1
959	32415	33700	3.8
963	33850	34400	1.6
991	45842	43400	5.6

From Table 5.3, we see that the system works well for illuminance values up to 6130 lx. In this range, the maximum error is 10.0%, with an average error of 4.5%. Above that range, the error goes up fast, to above 50% at 42600 lx. It makes sense that this happens, since this specific solar cell is made for indoor use, and its datasheet mentions a working range of up to 1000 lx. It is, therefore, interesting to note that the system works up to 6130 lx. For clarity, the error is plotted in Figure 5.1.

We run the same simulations using the data from Table 5.2. The results can be found in Table 5.4.

In Table 5.4 we see that the system also works for the Panasonic Amorton AM-5308, an outside type solar cell. The average error is 4.9% in a range from 600 lx up to 43400 lx. This is slightly higher but very close to the error of the Panasonic Amorton AM-1456. The maximum error is 9.8%, which is similar to the Panasonic Amorton AM-1456. Interestingly, the range over which this error stays low is much larger. The system works well from 600 lx up to 45000 lx. For clarity, the error is plotted in Figure 5.2.

The outdoor solar cell does not work well at illuminance lower than 600 lx. This intuitively makes some sense, as the solar cell is made to work at much higher illuminance. Although the datasheet does not specifically mention a working range, all graphs in the datasheet start at 5 klx.

We have made all measurements from Table 5.2 in sunlight (either direct or indirect). We have also made a

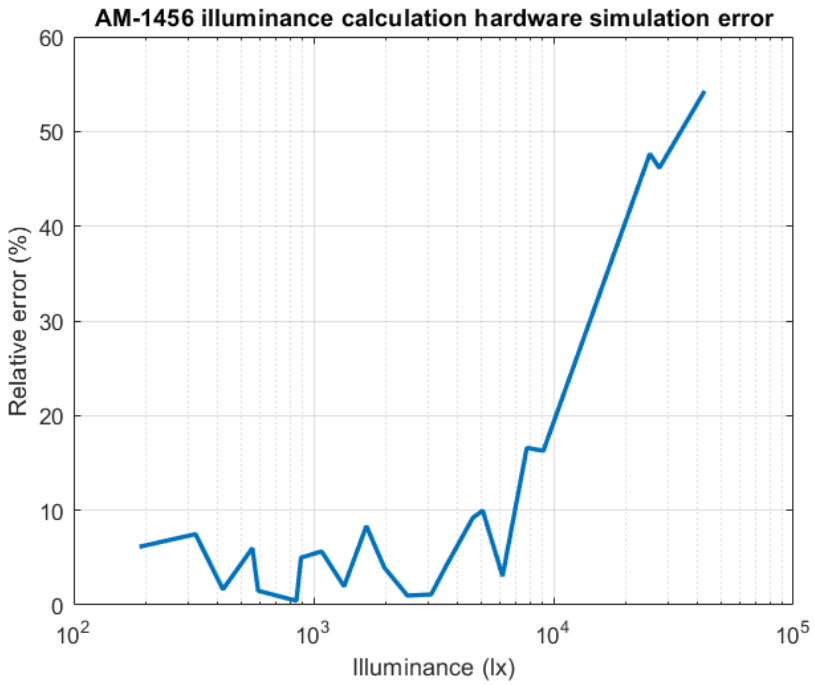

Figure 5.1: Illuminance calculation hardware simulation with Panasonic Amorton AM-1456 measured data

few measurements using indoor TL lighting. We simulate those results using both the same calibration as used in Table 5.4, but also with its own calibration. The results can be found in Tables 5.5 and 5.6 respectively.

Table 5.5: Simulation with measured solar cell data of Panasonic Amorton AM-5308 under TL lighting, with sunlight calibration

Actual illuminance (lx)	656	813	913
Simulated illuminance (lx)	445	571	602
Error (%)	32.2	29.8	34.0

Table 5.6: Simulation with measured solar cell data of Panasonic Amorton AM-5308 under TL lighting, with indoor calibration

Actual illuminance (lx)	656	813	913
Simulated illuminance (lx)	657	843	890
Error (%)	0.1	3.6	2.6

From these tables, we see that the system does not work indoors with outdoor calibration or the other way around. It has an error of about 30%. However, when we calibrate the solar cell for indoor use, the system seems to be working fine, with an error of about 3% over the small range it was measured at. To draw more conclusions regarding the system with different light sources, we would need more measurement data over a larger range of the solar cell with indoor-lighting data.

5.3 Circuit simulator verification

To verify if the illuminance calculation system works and how accurate it is, we simulate the system using Cadence Spectre, both with only the illuminance calculation hardware, and with the charge pump, the current sensor, and the ADC added.

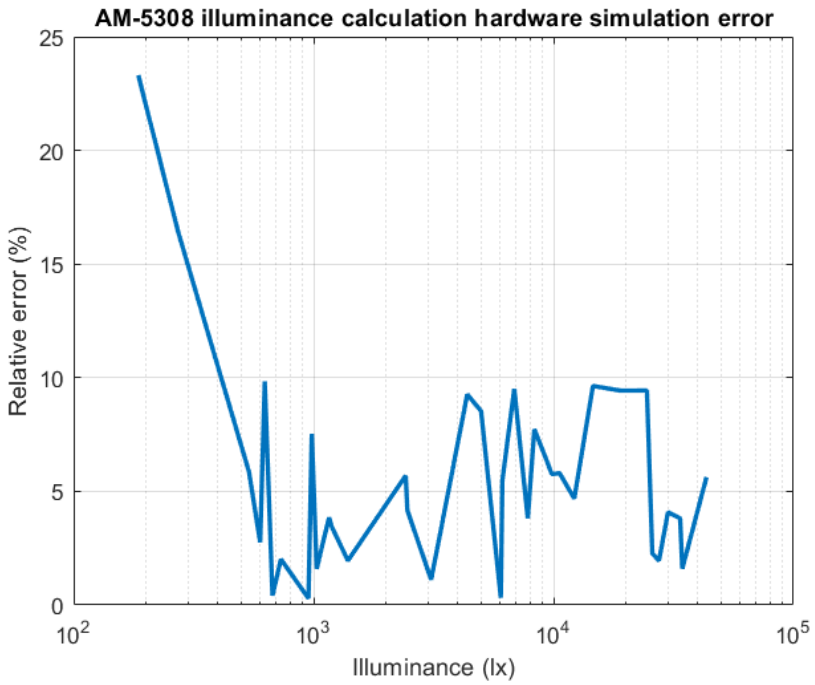

Figure 5.2: Illuminance calculation hardware simulation with Panasonic Amorton AM-5308 measured data

5.3.1 Illuminance hardware simulation

We have simulated the illuminance calculator by giving it a digital 8-bit input as the ADC would. Also, we inputted a digital reference, the control bits for the charge pump, and the control bits for the current mirror. By comparing the output of the illuminance hardware to the calculated output in the ideal case, we can see if the results are correct or not. Again, we can verify the results by checking if the output code matches the calculated code for each ADC input. The results table of the calculation can be found in Appendix E, and the full calculation Excel sheet can be found in Appendix F. After simulation, we export the data to MATLAB to calculate the output code from the output signals. Part of the simulation output can be found in Figure 5.3. In Figure 5.3 a, the input ADC code that is used as input to the illuminance calculation system is plotted. In Figure 5.3 b, the illuminance, in lux, as calculated by the system is plotted. In Figure 5.3 c the signal ”Done” is plotted. This signal is an output of the system and tells the user that the calculation is done.

Note that the calculation in the Excel file is without rounding, and thus the calculated illuminance as given in the Excel file is slightly different from the illuminance found by event-based simulation, as plotted in Figure 5.3 b. However, this error does stay within 2.5% at all times.

The spikes in Figure 5.3 c, are due to the translation step from volts to output code. Also, these spikes only appear at moments when the output is not valid yet, which is the case when the ”Done” signal is low.

The average error of the illuminance computation simulation compared to the theoretical perfect calculation is 0.4%, with a maximum error of 0.9%. This falls well within the needed margins.

With measured solar cell data We do the same simulation using the data from Tables 5.1 and 5.2. Since a digital system is simulated, we expect that the results will be exactly the same as given in Tables 5.3 and 5.4. Simulation shows that the results of the event-based simulation are indeed exactly the same as the results of the simulation in Cadence Spectre, with only the illuminance calculation hardware.

5.3.2 Chip simulation

Next, we simulate the illuminance computation hardware together with the important parts of the NOWI NH16 chip. This includes the charge pump, the current sensor, and the ADC. For the input, we made the four-parameter solar cell model in Cadence. The solar cell output is connected to the charge pump. The charge pump output

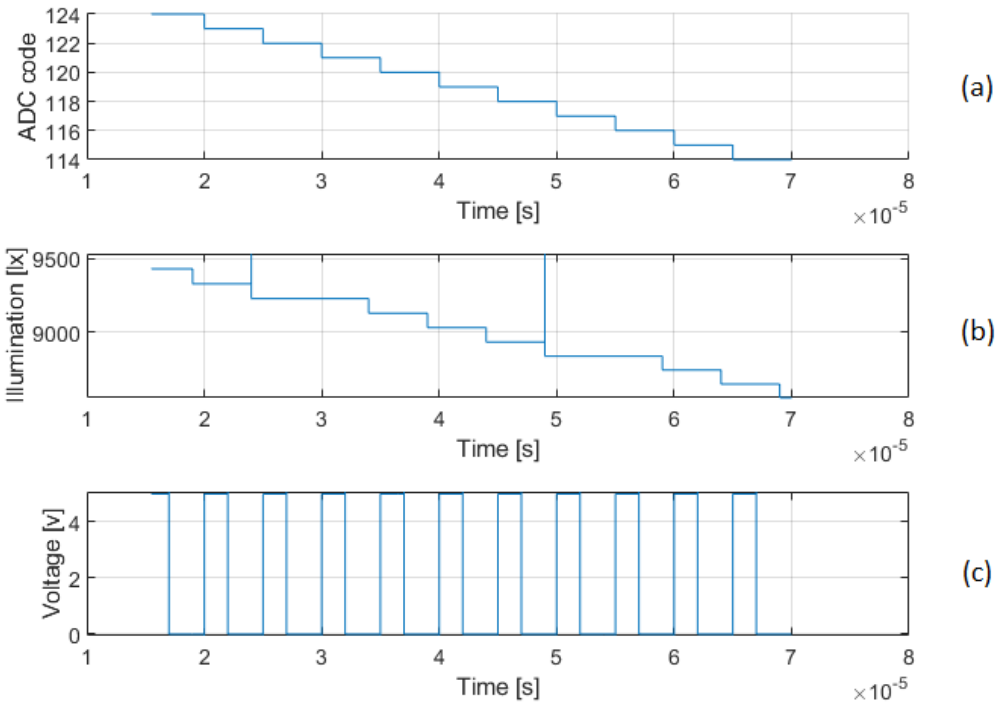

Figure 5.3: Simulation of the illuminance calculation hardware: (a) ADC code input, (b) Calculated illuminance output, (c) ”Done” signal

is connected to the current sensor. The current sensor creates a sense voltage via a sense resistor, which is connected to the ADC. The current sensor is also connected to a battery model. Lastly, the ADC output, which is an 8-bit binary signal, is connected to the illuminance calculation hardware. The setup can be seen in Figure 5.4. To save simulation time, we set the settings for the charge pump and current sensor manually.

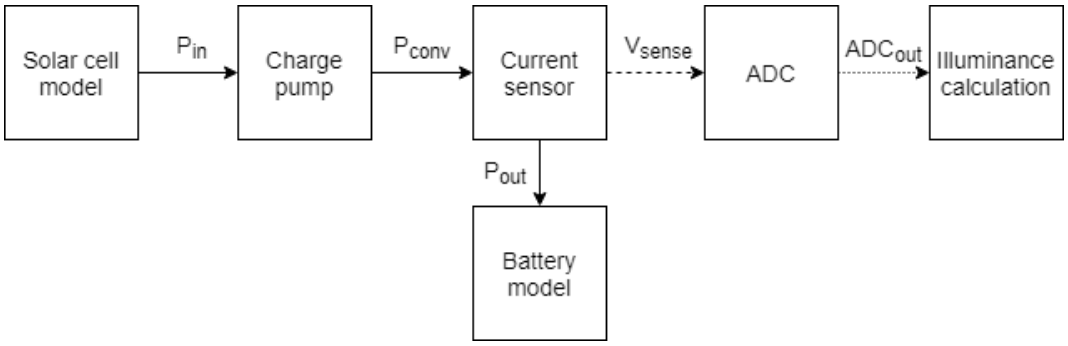

Figure 5.4: Diagram of the chip simulation setup

The parameters for the solar cell model are varied to vary the input current to the charge pump. We try to get this input current as close as possible to the measured input currents, which can be found in Tables 5.1 and 5.2. It is important to simulate until the output voltage of the current sensor is settled. If the input current is not exact we use interpolation to find the correct ADC value for each measured value. We do this by using a linear interpolation between the two closest current values and calculating the expected current sensor output voltage. This voltage is then used as input in a simulation with the ADC, which gives the wanted ADC output value.

For this simulation, the results are not the same as in Tables 5.3 and 5.4, since the addition of the charge pump, the current sensor and the ADC introduce errors. The results for the indoor and outdoor solar cells can be

found in Tables 5.7 and 5.8 respectively. The error is the relative error between the measured illuminance and the calculated illuminance.

Table 5.7: Chip simulation with measured data from Panasonic Amorton AM-1456

Digital output	Calculated illuminance (lx)	Measured illuminance (lx)	Error (%)
454	137	188	27.3
510	251	322	22.2
545	366	419	12.7
581	540	556	2.8
581	540	587	7.9
617	798	848	5.9
617	798	887	10.0
645	1081	1080	0.1
659	1258	1340	6.1
689	1741	1660	4.9
714	2282	1970	15.9
728	2656	2460	8.0
752	3444	3090	11.5
757	3636	3550	2.4
775	4419	4610	4.2
783	4819	5070	5.0
812	6597	6130	7.6
815	6814	7740	12.0
827	7760	9070	14.4

The relative error plotted against measured illuminance can be found in Figures 5.5 and 5.6 for the indoor en outdoor solar cells respectively.

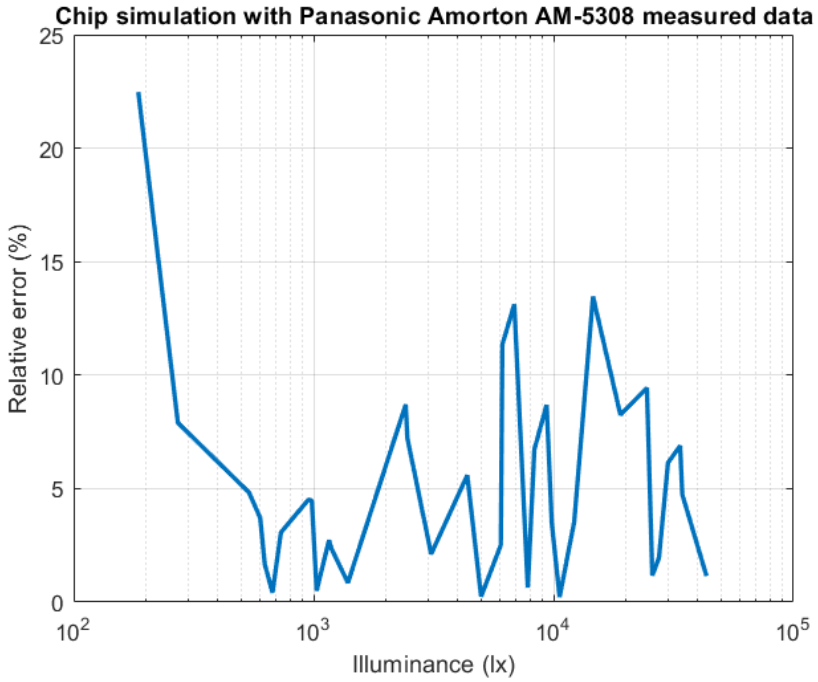

Figure 5.6: Chip simulation with Panasonic Amorton AM-5308 measured data

For the indoor solar cell, the average error is 9.1% with a maximum error of 27.3%, over a range from 188 lx to 6130 lx. In Figure 5.5 we see that the error is especially high for small illuminance values. This is caused by the charge pump, which has a lower efficiency for lower input power. From powers of about 250 μW and up, the

Table 5.8: Chip simulation with measured data from Panasonic Amorton AM-5308

Digital output	Calculated illuminance (lx)	Measured illuminance (lx)	Error (%)
459	144	186	22.5
510	251	272	7.9
576	512	538	4.8
587	577	599	3.7
593	616	626	1.7
601	671	674	0.4
606	709	731	3.1
629	909	952	4.5
632	939	983	4.5
641	1035	1030	0.5
654	1192	1160	2.7
656	1218	1190	2.3
669	1402	1390	0.9
711	2209	2420	8.7
714	2282	2460	7.2
740	3025	3090	2.1
779	4614	4370	5.6
786	4978	4990	0.2
806	6182	6030	2.5
815	6814	6120	11.3
827	7760	6860	13.1
827	7760	7810	0.6
827	7760	8320	6.7
852	10173	9360	8.7
852	10173	9830	3.5
856	10624	10600	0.2
872	12634	12200	3.6
872	12634	14600	13.5
917	20568	19000	8.3
924	22188	24500	9.4
939	26102	25800	1.2
942	26964	27500	1.9
946	28158	30000	6.1
956	31379	33700	6.9
960	32768	34400	4.7
987	43898	43400	1.1

charge pump has an efficiency of more than 91%. This slowly drops off to about 80% at $80\ \mu$W, but below $50\ \mu$W the efficiency falls under 70%, and even down to below 50% at $30\,\mu$W. This explains why in this simulation the error for the lowest illuminance values is larger than in the simulation with only the illuminance calculation hardware.

If we disregard the first two values, we find an average error of 7.0% with a maximum error of 15.9% over a range from 419 lx to 6130 lx.

For the outdoor solar cell, the average error is 4.7% with a maximum error of 13.5%, over a range from 272 lx to 43400 lx. What stands out is that the error at 272 lx, 7.9%, is much lower in this simulation than in the event-based simulation, 16.5%. This seems to be a case where, coincidentally, two errors, solar cell efficiency and charge-pump efficiency, compensate each other, which leads to a better result.

5.4 FPGA verification

To verify the design, we made a prototype of the system on an FPGA, a Zybo Z7. The NOWI NH16 uses I2C to communicate, so the FPGA will use I2C too. We sent the ADC output, the charge pump control bits, and the current mirror control bits to the FPGA via I2C. On the chip, this I2C communication is not necessary as the system

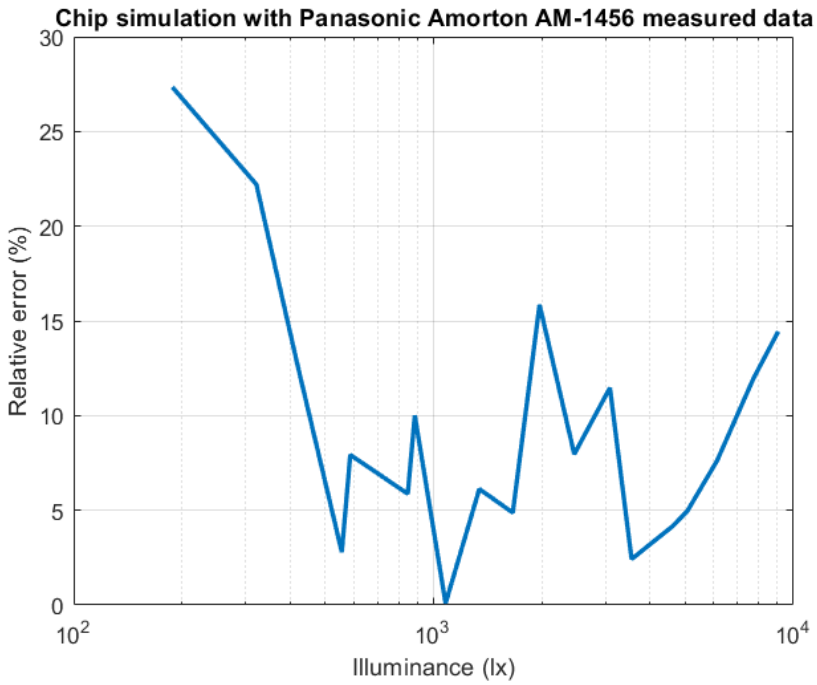

Figure 5.5: Chip simulation with Panasonic Amorton AM-1456 measured data

will read its data directly from the registers. To interface with the FPGA, we use an Arduino UNO. The Arduino can communicate with the FPGA over I2C, and can then communicate with a PC via serial USB, to display results.

To include I2C communication we add two blocks. The first one is an I2C Master, which controls the actual bits on the SDA and SCL lines to make I2C communication happen. The other block is an I2C controller, which controls the I2C Master block by setting the address, data bits, read-write bit, enable bit, and reset bit of the I2C Master.

A diagram of the FPGA setup can be found in Figure 5.7. The Verilog files can be found in Appendix D. We did not change the two LUT files compared to the chip version, and they can be found in Appendix B.

An I2C Master is not something new, and many have already been made in different hardware description languages, thus we used an existing one, specifically the I2C Master v2.2 from Digikey [78].

The I2C control block consists of an FSM controlling the values of the I2C Master. It sets the address, and then can both read from or write to the Arduino. The data read is saved, and then the LUTs use the ADC output, the charge-pump control bits, and current mirror control bits, to give the correct data to the illuminance calculation block. This block then calculates the illuminance and calibrates the system if that has not happened yet. The system subsequently sends the result back to the Arduino.

We use a button on the FPGA to make the FPGA request the reference value from the Arduino. After receiving the value, the FPGA will recalibrate itself with the newly inputted value.

Since the NOWI NH16 is not yet finished, in this version we do not use the NH16 yet, however, once the design is finished and the chip has been fabricated, it is possible to rebuild the design to communicate with the NH16 as well. The system will then read ADC data and control bits directly from the NH16, and use those to verify the system. Instead, in this version the Arduino sends the data to the FPGA, from the datasets as given in Tables 5.1 and 5.2.

To verify if the system works correctly, the system compares the output of the FPGA to the data obtained in the event-based simulation, and the circuit simulator simulation without the charge pump, current sensor, and ADC, which indeed is the case. These results can be found in Tables 5.3 and 5.4.

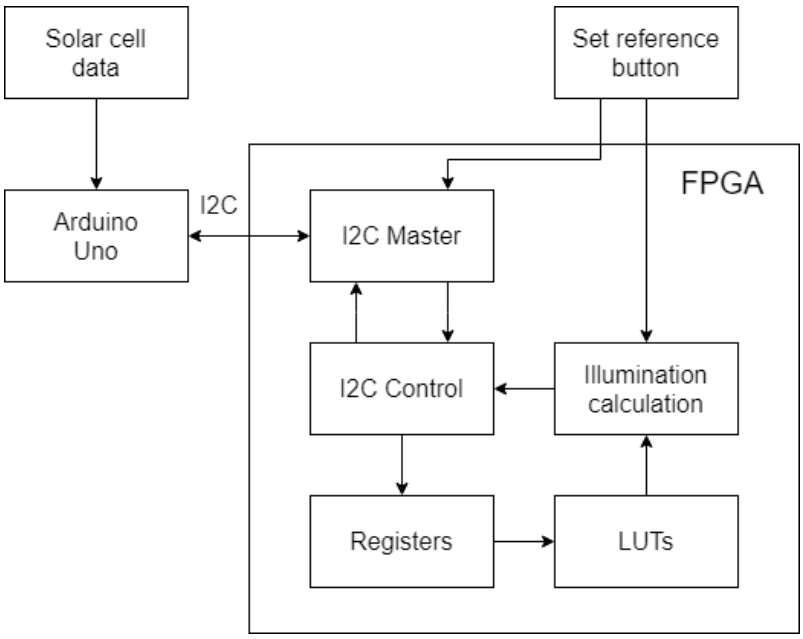

Figure 5.7: FPGA verification setup diagram

Table 5.9: Comparison of energy harvesting sensing systems

Reference	Sensing	Inaccuracy	Transducer	Working at MPP	Calibration possible	Power usage	Range
[10]	Illuminance	20%	Solar cell	No	Yes	$> 6\ \mu$W	25 - 25000 lx
[11]	Illuminance	N.A.	Solar cell	No	No	All power harvested	N.A.
[18]	Illuminance	N.A.	Solar cell	No	No	N.A.	N.A.
[12]	Sun angle	11%	Photodiodes	No	No	N.A.	360°
[15]	Pixels	N.A.	Photodiodes	No	No	Harvested power + $4\ \mu$W	N.A.
[19]	Pixels	N.A.	Photodiodes	During harvesting	No	10 mW	N.A.
[20]	Pixels	N.A.	Photodiodes	During harvesting	No	N.A.	N.A.
This work (indoor/ outdoor)	Illuminance	7.0% 4.7%	Solar cell	Yes	Yes	$121.68\ \mu$W at 1 kHz	419 - 6130 lx 272 - 43400 lx

A comparison of this thesis with the most similar works from Section 2.1 can be found in Table 5.9. From this table, we can see that this project is separated from similar works, because it is the only one that works at MPP during both harvesting and sensing. Compared to the works from which accuracy was estimated, this project seems to have good accuracy. Also, the possibility to calibrate the system is an advantage of this project that many of the other works do not have.

5.5 Power consumption

To get an idea of the power consumption of the system, we simulate the illuminance computation hardware and calculated the average current consumption of the system. The simulation is done with a 1 kHz clock and input values changing every 10 ms, as is the fastest we can expect them to change in the NOWI NH16. We let Cadence calculate the average supply current over a period of 1 s. The result is 27.04 μA, with a power supply of 4.5 V, which results in average power consumption of $121.68\,\mu$W. This is much less than the power predicted by Cadence Genus in Section 4.4.2, which makes sense because in Section 4.4.2, Cadence Genus assumed that the system works continuously, while in reality it only needs to do a conversion once every 10 ms.

6. Conclusion

6.1 Conclusions

We created an algorithm to calculate illuminance from solar cell current, with the solar cell working at maximum power point. Also, we made a calibration algorithm so that the system can work with any solar cell, and compensate for gain errors in the chip.

We implemented the algorithm in Verilog, which we synthesised with Cadence Genus to implement on-chip.

We verified the system via event-based simulation in ModelSim, and circuit simulation in Cadence.

Simulating only the calculation hardware shows that, compared to a theoretical ideal calculation of the resulting illuminance from any ADC input, the hardware calculates the illuminance from an ADC input between 0-255 with an error of 0.4% on average, with a maximum error of 0.9%.

Simulation of measured solar cell data shows an error of 4.5% on average with a maximum error of 10.0% for an indoor solar cell over a range of up to 6130 lx, and an average error of 4.9% with a maximum error of 9.8% for an outdoor solar cell over a range from 599 lx up to 43400 lx. These results were obtained for simulations via event-based simulation in ModelSim and simulation of the illuminance calculation hardware using Cadence.

Also, we made an FPGA implementation of the circuit, and we simulated it with measured solar cell data. This reached the same results as in event-based simulation, via simulation in Cadence, and on the FPGA, confirming the functionality of the circuit.

Lastly, we validated the system using Cadence with the charge pump, the current sensor, and the ADC connected. In this simulation, we obtained an average error of 7.0% with a maximum error of 15.9% over a range from 419 lx to 6130 lx for the indoor solar cell, and an average error of 4.7% with a maximum error of 13.5%, over a range from 272 lx to 43400 lx for the outdoor solar cell.

6.2 Main contributions

To discuss the main contributions discussed in this project, we recall the research question:

Can a low-power energy harvester MPPT be used as a sensor front end for illuminance sensing?

In this project, we showed how an energy harvester MPPT can be used as an illuminance sensing system simultaneously while harvesting energy. Firstly, we did extensive research to see what errors can be expected, and which of them have the biggest impact. The biggest error sources in this project appear in the solar cell, due to variation in light source or temperature. In the NOWI NH16, the ADC and the charge pump cause the largest errors.

Also, we investigated what sensing methods can be used in this project, and what accuracy to expect from them. In the end, we determined that the senseFET system, as implemented in the NOWI NH16, is the best sys-

tem to use for this project.

We investigated different ways to implement a calculation algorithm, and we made a trade-off between them. In the end, we developed a calculation algorithm that only uses the MPP current as an input to calculate the illuminance. This way the MPPT can work at MPP at all times, and thus the illuminance calculation hardware has no impact on the workings of the energy harvester. It is, therefore, possible to integrate this design into the NOWI NH16 without a lot of modifications.

Also, since no specific solar cell is mentioned, it is important that the developed system can work with any solar cell connected to the system. To do so, we introduced a calibration system allowing the system to be calibrated for any solar cell that can be connected to the energy harvester. Besides calibrating for the differences between solar cells, the calibration system is also important for the energy harvester itself, since its relative accuracy is good, but its absolute accuracy is not good enough without calibration.

Lastly, we implemented the hardware for the illuminance calculation and calibration algorithm in Verilog, and we synthesised them, both for implementation on-chip, or in Cadence, and for implementation on an FPGA. The circuit for the on-chip implementation is made specifically for integration with the NOWI NH16.

We verified the design with measured data of two solar cells, namely the Panasonic Amorton AM-1456 and the Panasonic Amorton AM-5308. The first one is made for indoor use, while the latter one is meant for outdoor use. This shows the design works, not only for one specific (type of) solar cell, but it shows the validity of the design with other solar cells as well.

6.3 Recommendations

6.3.1 FPGA simulation with NH16

The main point of the FPGA implementation was to test the entire circuit, with a solar cell and the NOWI NH16, and use the FPGA to simulate the calculation hardware. This way, the entire implementation of the design can be tested at once, which is the best way to verify the functioning of the design. However, since the NH16 is not yet finished, it was not possible to make this implementation. To verify the accuracy of the entire system in combination with the NOWI NH16, it is interesting to test this implementation once the NH16 is finished. An implementation for this can easily be made using the Verilog files made for this project. Only some small adjustments to which I2C messages are read are necessary. The files can be found in Appendix D.

6.3.2 Calibration improvement

At the moment the system is built to be calibrated at a single point, which in theory should work, and as shown in this project, is good enough in practice. However, it can be difficult to get the perfect measurement to calibrate the system in one go. Looking at Table 5.3, we can calculate the constant part for each value, which represents $\frac{G_0 \times D \times C}{I_{sc0}}$, in which D is the ratio between I_{sc} and I_{mpp}. Calculating this constant part for each value gives a range between 1.48 and 1.76. The best calibration value then is somewhere in the middle. However, if you use a measurement that is not carefully conducted, the calibration parameter might end up at one of the extremes. This would result in a bigger average error over the illuminance range. Instead, an improvement could be to calibrate using multiple measurements, if possible at multiple illuminance points, which would average out an error in the calibration variable.

6.3.3 Other solar cells

We verified the design using two different types of solar cells, however, even though the solar cells have a different purpose, indoor versus outdoor, they are of the same brand and family, Panasonic Amorton. Therefore, for future work, testing solar cells of different brands is advised, to confirm if the system works equally well with a variety of solar cells.

6.3.4 Temperature correction

In the measurement data, we did not measure the temperature. It is expected that the temperature has varied only a little between measurements. The effect temperature has on the calculated illuminance has been discussed throughout the project and we expect it to have only a small effect on the calculated illuminance. It is possible to add temperature correction to the system to suppress this error. For this, a temperature sensor needs to be added. This would make the calculation slightly more complex. The calculation that then needs to be done is according to Equation 6.1.

$$G = \frac{G_0 I_{sc}}{I_{sc0}[1 + \alpha(T - T_0)]} \tag{6.1}$$

In this case, another LUT is needed to convert the temperature factor to a logarithmic number. First $1+\alpha(T-T_0)$ needs to be calculated, and then a LUT can be used to transform the resulting number into a logarithmic number. Especially in more extreme conditions, at temperatures close to 0°C or above 40°C, this correction could make the system more accurate. Disadvantages are that adding a temperature sensor requires extra area and draws extra power.

6.3.5 Light source variation

In this project, the light source used was mostly the sun. Only for a few measurements, we used indoor TL-lighting. For the indoor solar cell, the different light sources did not seem to matter much, however, the outdoor solar cell gives very different results with indoor lighting. Other than a very small illuminance range, we did not gather data with indoor lighting. For future work, it can be interesting to calibrate the system with one light source and determine the error when illuminated by another light source. It would then also be possible to find the best calibration setting for different illuminance ranges. For higher illuminance, the light source can only be the sun. For lower illuminance, the light source can be both sun or indoor lighting. For different ranges, it is thus possible to find a more optimal calibration parameter than just one over the entire range. This corrects for the light source that is expected, or gives the best estimate if multiple light sources are possible.

Bibliography

[1] Maxim Integrated, *Low-Power Digital Ambient Light Sensor with Enhanced Sensitivity*, 2011.

[2] Texas Instruments, *OPT3001 Ambient Light Sensor (ALS)*, 2014. https://www.ti.com/lit/ds/symlink/opt3001.pdf?ts=1626708316190&ref_url=https%253A%252F%252Fwww.google.com%252F.

[3] Yokogawa, *510 Series Digital Lux Meter*, 2013. https://cdn.tmi.yokogawa.com/BU510-EN.pdf.

[4] Y. L. Sum, S.-C. Chien, Y. W. E. Chan, B. H. Soong, and K. J. Tseng, "High accuracy and resolution illuminance sensing for intelligent energy management," in *2015 IEEE International Telecommunications Energy Conference (INTELEC)*, pp. 1–6, 2015.

[5] A. Fish, S. Hamami, and O. Yadid-Pecht, "Self-powered active pixel sensors for ultra low-power applications," in *2005 IEEE International Symposium on Circuits and Systems*, pp. 5310–5313 Vol. 5, 2005.

[6] A. Pandharipande and S. Li, "Light-harvesting wireless sensors for indoor lighting control," *IEEE Sensors Journal*, vol. 13, no. 12, pp. 4599–4606, 2013.

[7] Broadcom / Avago, *APDS-9004 Miniature Surface-Mount Ambient Light Photo Sensor*, 2006. https://www.mouser.com/datasheet/2/38/V01-0184EN-107079.pdf.

[8] C. M. Nguyen, J. Mays, D. Plesa, S. Rao, M. Nguyen, and J.-C. Chiao, "Wireless sensor nodes for environmental monitoring in internet of things," in *2015 IEEE MTT-S International Microwave Symposium*, pp. 1–4, 2015.

[9] Naveen, H. Pasupuleti, R. Rao, D. Selvakumar, Srinivasan, and S. R. K. Reddy, "Energy aware self powered wireless sensor mote," in *2012 Sixth International Conference on Sensing Technology (ICST)*, pp. 630–636, 2012.

[10] S. Yamada and H. Toshiyoshi, "An illuminance sensor integrated with analog digital converter using pulse density modulation," in *2017 IEEE SENSORS*, pp. 1–3, 2017.

[11] K. Kadirvel and J. Carpenter, "Self-powered, ambient light sensor using bq25504," 2013.

[12] H. Wu, A. Emadi, G. de Graaf, J. Leijtens, and R. F. Wolffenbuttel, "Self-powered sun sensor microsystems," *Procedia Chemistry*, vol. 1, no. 1, pp. 1363 – 1366, 2009. Proceedings of the Eurosensors XXIII conference.

[13] H. Song, Z. Lu, T. Luo, J. B. Christen, and H. Wang, "A cmos self-powered monolithic light direction sensor with sar adc," in *2014 27th IEEE International System-on-Chip Conference (SOCC)*, pp. 58–62, 2014.

[14] K. Kang, K. Na, D. Kwon, J.-Y. Lee, and I. Park, "Self-powered gas sensor using thin-film photovoltaic cell and microstructured colorimetric film," in *2017 19th International Conference on Solid-State Sensors, Actuators and Microsystems (TRANSDUCERS)*, pp. 1536–1539, 2017.

[15] C. Shi, M. K. Law, and A. Bermak, "A novel asynchronous pixel for an energy harvesting cmos image sensor," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 1, pp. 118–129, 2011.

[16] Leviton, *WSCPC-W*, 2010. https://www.leviton.com/en/docs/LevNet_RF_Light_Sensor_%28WSCPC%29.pdf.

[17] Molex, *180997-0006*, 2019. https://www.molex.com/molex/products/part-detail/molex_parts/1809970006.

[18] A.-K. Shedletsky, E. G. de Jong, F. R. Rothkopf, and A. S. Montevirgen, “Solar cell ambient light sensors for electronic devices,” U.S. Patent US20140152632A1, Apr. 2012.

[19] K. Z. Ahmed, M. F. Amir, J. H. Ko, and S. Mukhopadhyay, “Reconfigurable 96×128 active pixel sensor with 2.1µw/mm2 power generation and regulated multi-domain power delivery for self-powered imaging,” in *ESSCIRC Conference 2016: 42nd European Solid-State Circuits Conference*, pp. 507–510, 2016.

[20] W. D. Leon-Salas, T. Fischer, X. Fan, G. Moayeri, and S. Luo, “A 64×64 image energy harvesting configurable image sensor,” in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1914–1917, 2016.

[21] Y. Umetsu, Y. Nakamura, Y. Arakawa, M. Fujimoto, and H. Suwa, “Ehaas: Energy harvesters as a sensor for place recognition on wearables,” in *2019 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pp. 1–10, 2019.

[22] H. P. Forghani-zadeh and G. A. Rincon-Mora, “Current-sensing techniques for dc-dc converters,” in *The 2002 45th Midwest Symposium on Circuits and Systems, 2002. MWSCAS-2002.*, vol. 2, pp. II–II, 2002.

[23] S. Singh, D. Mandal, B. Bakkaloglu, and S. Kiaei, “Low-power/low-voltage integrated cmos sense resistor-free analog power/current sensor compatible with high-voltage switching dc–dc converter,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 6, pp. 2208–2218, 2019.

[24] S. Ziegler, R. C. Woodward, H. H. Iu, and L. J. Borle, “Current sensing techniques: A review,” *IEEE Sensors Journal*, vol. 9, no. 4, pp. 354–376, 2009.

[25] S. H. Shalmany, D. Draxelmayr, and K. A. A. Makinwa, “A micropower battery current sensor with $\pm$ 0.03% (3σ) inaccuracy from -40 to +85°c,” in *2013 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, pp. 386–387, 2013.

[26] C. Liu and H. J. Liu, “Offset error reduction in open loop hall effect current sensors powered with single voltage source,” in *2014 IEEE International Workshop on Applied Measurements for Power Systems Proceedings (AMPS)*, pp. 1–6, 2014.

[27] P. Givelin, M. Bafleur, E. Tournier, T. Laopoulos, and S. Siskos, “Application of a cmos current mode approach to on-chip current sensing in smart power circuits,” *IEE Proceedings - Circuits, Devices and Systems*, vol. 142, no. 6, pp. 357–363, 1995.

[28] S. Rao, Q. Khan, S. Bang, D. Swank, A. Rao, W. McIntyre, and P. K. Hanumolu, “A 1.2-a buck-boost led driver with on-chip error averaged sensefet-based current sensing technique,” *IEEE Journal of Solid-State Circuits*, vol. 46, no. 12, pp. 2772–2783, 2011.

[29] E. N. Stefanov, D. Zupac, and E. de Frésart, “Current sense dynamics during turn-on of power mosfet,” in *2014 IEEE 26th International Symposium on Power Semiconductor Devices IC’s (ISPSD)*, pp. 386–389, 2014.

[30] V. Michal, “Absolute value, 1% linear and lossless current-sensing circuit for the step-down dc-dc converters with integrated power stage,” *IEEE Journal of Solid-State Circuits*, vol. 49, pp. 1256 – 1270, 05 2015.

[31] H. Lavric and R. Fiser, “Lossless current sensing technique on mosfet rds(on) with improved accuracy,” *Electronics Letters*, vol. 46, no. 5, pp. 370–371, 2010.

[32] O. Lopez-Lapena, M. T. Penella, and M. Gasulla, “A new mppt method for low-power solar energy harvesting,” *IEEE Transactions on Industrial Electronics*, vol. 57, no. 9, pp. 3129–3138, 2010.

[33] K. Itoh, M. Muraguchi, and T. Endoh, “High accurate and low loss current sensing method with novel current path narrowing method for dc-dc converters and its demonstration,” in *2016 IEEE International Telecommunications Energy Conference (INTELEC)*, pp. 1–6, 2016.

[34] Yang Zhang, R. Zane, A. Prodic, R. Erickson, and D. Maksimovic, “Online calibration of mosfet on-state resistance for precise current sensing,” *IEEE Power Electronics Letters*, vol. 2, no. 3, pp. 100–103, 2004.

[35] S. Tian, W. Ballar, X. Chen, Y. Yan, Z. Liu, R. Ying, and E. Beville, “A novel dcr current sensing scheme for accurate current readback in power umodule applications,” in *2020 IEEE Applied Power Electronics Conference and Exposition (APEC)*, pp. 2763–2765, 2020.

[36] S. Hainz, E. de la Torre, and J. Güttinger, “Comparison of magnetic field sensor technologies for the use in wheel speed sensors,” in *2019 IEEE International Conference on Industrial Technology (ICIT)*, pp. 727–731, 2019.

[37] P. Midya, P. T. Krein, and M. F. Greuel, “Sensorless current mode control-an observer-based technique for dc-dc converters,” *IEEE Transactions on Power Electronics*, vol. 16, no. 4, pp. 522–526, 2001.

[38] R. Channappanavar and S. Mishra, “A novel current estimation technique for digital controlled switching converters operating in ccm and dcm,” in *2017 IEEE Energy Conversion Congress and Exposition (ECCE)*, pp. 1781–1786, 2017.

[39] W. Xian, X. Li, D. F. Wang, T. Kobayashi, T. Itoh, and R. Maeda, “Precise current sensing using a piezoelectric cantilever based current sensor,” in *2017 19th International Conference on Solid-State Sensors, Actuators and Microsystems (TRANSDUCERS)*, pp. 1057–1060, 2017.

[40] E. S. Leland, C. T. Sherman, P. Minor, R. M. White, and P. K. Wright, “A new mems sensor for ac electric current,” in *SENSORS, 2010 IEEE*, pp. 1177–1182, 2010.

[41] S. Fan, J. Dong, R. Zhang, Y. Zhao, and L. Geng, “A low-power 0.8% sensing error on-chip current sensor with auto-accuracy adjustment,” in *2018 IEEE International Conference on Integrated Circuits, Technologies and Applications (ICTA)*, pp. 106–107, 2018.

[42] W. Li, X. Yin, D. Chen, W. Chen, Z. Zhang, and Y. Xia, “The study of transient performance of current sensor based on rogowski coil and its application in dynamic simulation experiment,” in *2006 International Conference on Power System Technology*, pp. 1–6, 2006.

[43] W. Xian and D. F. Wang, “Position and orientation correction scheme for current sensing based on magnetic piezoelectric cantilevers,” *Applied Physics Letters*, vol. 110, no. 14, p. 143501, 2017.

[44] J. Huang, X. Wu, and X. Wang, “A noninvasive ac current sensor with permanent-magnet biased pzt cantilever,” in *2015 IEEE SENSORS*, pp. 1–4, 2015.

[45] S. Singh, D. Mandal, B. Bakkaloglu, and S. Kiaei, “Sense resistor-free analog power sensor for boost converter with 14.1% gain error and 9.4% offset error,” in *2018 IEEE 9th Latin American Symposium on Circuits Systems (LASCAS)*, pp. 1–4, 2018.

[46] S. Song, “Design of a low-current voltage dividers,” 2011.

[47] W. Tress, M. Yavari, K. Domanski, P. Yadav, B. Niesen, J. P. Correa Baena, A. Hagfeldt, and M. Graetzel, “Interpretation and evolution of open-circuit voltage, recombination, ideality factor and subgap defect states during reversible light-soaking and irreversible degradation of perovskite solar cells,” *Energy Environ. Sci.*, vol. 11, pp. 151–165, 2018.

[48] S. M. Alghuwainem, “Matching of a dc motor to a photovoltaic generator using a step-up converter with a current-locked loop,” *IEEE Transactions on Energy Conversion*, vol. 9, no. 1, pp. 192–198, 1994.

[49] M. A. G. de Brito, L. Galotto, L. P. Sampaio, G. d. A. e Melo, and C. A. Canesin, “Evaluation of the main mppt techniques for photovoltaic applications,” *IEEE Transactions on Industrial Electronics*, vol. 60, no. 3, pp. 1156–1167, 2013.

[50] Panasonic, *Amorton AM-5610*, 2014. `https://nl.mouser.com/datasheet/2/315/panasonic_AM-5610CAR-1196922.pdf`.

[51] IXOLAR, *KXOB25-14X1F*, 2010. `https://waf-e.dubudisk.com/anysolar.dubuplus.com/techsupport@anysolar.biz/O18Ae08/DubuDisk/www/Gen3/KXOB25-14X1F%20DATA%20SHEET%20202007.pdf`.

[52] A. McLaren and K. Martin, “Generation of accurate on-chip time constants and stable transconductances,” *IEEE Journal of Solid-State Circuits*, vol. 36, no. 4, pp. 691–695, 2001.

[53] X. Zhang, B. Ni, I. Mukhopadhyay, and A. B. Apsel, “Improving absolute accuracy of integrated resistors with device diversification,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, no. 6, pp. 346–350, 2012.

[54] M. D. Seeman and S. R. Sanders, “Analysis and optimization of switched-capacitor dc–dc converters,” *IEEE Transactions on Power Electronics*, vol. 23, no. 2, pp. 841–851, 2008.

[55] M. D. Seeman, “Analytical and practical analysis of switched-capacitor dc–dc converters,” *Berkeley, CA, Tech. Rep. EECS-2006-11*, 2006.

[56] R. dang Yang, “A high-precision capacitive sensor system for displacement measurements,” 2015.

[57] E. Raguvaran, N. Deepak Prasath, J. Alexander, N. Prithiviraj, and M. Santhanalakshmi, “A very-high impedance current mirror for bio-medical applications,” in *2011 IEEE Recent Advances in Intelligent Computational Systems*, pp. 828–830, 2011.

[58] S. S. Rajput, P. Vajpayee, and G. K. Sharma, “1v high performance current mirror for low voltage analog and mixed signal applications in submicron regime,” in *TENCON 2009 - 2009 IEEE Region 10 Conference*, pp. 1–4, 2009.

[59] C. Liu, “27.4 a 0.35mw 12b 100ms/s sar-assisted digital slope adc in 28nm cmos,” in *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, pp. 462–463, 2016.

[60] J. Luo, J. Li, N. Ning, Y. Liu, and Q. Yu, “A 0.9-v 12-bit 100-ms/s 14.6-fj/conversion-step sar adc in 40-nm cmos,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 10, pp. 1980–1988, 2018.

[61] H. Banba, H. Shiga, A. Umezawa, T. Miyaba, T. Tanzawa, S. Atsumi, and K. Sakui, “A cmos bandgap reference circuit with sub-1-v operation,” *IEEE Journal of Solid-State Circuits*, vol. 34, no. 5, pp. 670–674, 1999.

[62] R. Akshaya and S. Y. Siva, “Design of an improved bandgap reference in 180nm cmos process technology,” in *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT)*, pp. 521–524, 2017.

[63] L. Cristaldi, M. Faifer, M. Rossi, and S. Toscani, “An improved model-based maximum power point tracker for photovoltaic panels,” *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 1, pp. 63–71, 2014.

[64] V. Ng and S. Sanders, “A 92%-efficiency wide-input-voltage-range switched-capacitor dc-dc converter,” in *2012 IEEE International Solid-State Circuits Conference*, pp. 282–284, 2012.

[65] Panasonic, *Amorton AM-1456*, 2014. `https://nl.mouser.com/datasheet/2/315/panasonic_AM-1456CA-1196920.pdf`.

[66] W. De Soto, S. Klein, and W. Beckman, “Improvement and validation of a model for photovoltaic array performance,” *Solar Energy*, vol. 80, no. 1, pp. 78 – 88, 2006.

[67] D. Sera, R. Teodorescu, and P. Rodriguez, “Pv panel model based on datasheet values,” in *2007 IEEE International Symposium on Industrial Electronics*, pp. 2392–2396, 2007.

[68] A. Chatterjee, A. Keyhani, and D. Kapoor, “Identification of photovoltaic source models,” *IEEE Transactions on Energy Conversion*, vol. 26, no. 3, pp. 883–889, 2011.

[69] W. De Soto, “Improvement and validation of a model for photovoltaic array performance,” *M.S. Thesis, Solar Energy Laboratory, University of Wisconsin-Madison, USA*, 2004.

[70] E. Scolari, F. Sossan, and M. Paolone, “Photovoltaic model-based solar irradiance estimators: Performance comparison and application to maximum power forecasting,” *CoRR*, vol. abs/1705.04132, 2017.

[71] L. Cristaldi, M. Faifer, M. Rossi, and F. Ponci, “A simple photovoltaic panel model: Characterization procedure and evaluation of the role of environmental measurements,” *IEEE Transactions on Instrumentation and Measurement*, vol. 61, no. 10, pp. 2632–2641, 2012.

[72] Panasonic, *Amorton AM-1454*, 2014. `https://nl.mouser.com/datasheet/2/315/panasonic_AM-1454CA-1197022.pdf`.

[73] PowerFilm Solar, *LL200-2.4-37*, 2020. `https://www.powerfilmsolar.com/markets/ll200-24-37`.

[74] R. S. Hongal and D. Anita, “Comparative study of different division algorithms for fixed and floating point arithmetic unit for embedded applications,” 2016.

[75] T.-J. Kwon and J. Draper, “Floating-point division and square root using a taylor-series expansion algorithm,” *Microelectronics Journal*, vol. 40, no. 11, pp. 1601–1605, 2009. International Conference on Microelectronics Digital and Mixed-Signal Circuits and Systems.

[76] A. Mansour, A. El-Sawy, M. Ismail, and A. Sayed, “A new hardware implementation of base 2 logarithm for fpga,” *International Journal of Signal Processing Systems*, vol. 3, 01 2014.

[77] Panasonic, *Amorton AM-5308*, 2008. `https://pdf1.alldatasheet.com/datasheet-pdf/view/251224/SANYO/AM-5308.html`.

[78] Digikey, *I2C Master (VHDL)*, 2012. `https://forum.digikey.com/t/i2c-master-vhdl/12797`.

A.
MPP linearisation

The measuring device includes an MPPT, and it is, therefore, reasonable to assume the solar cell works close to its MPP. If we can make a linearisation around the MPP, we can reduce the calculation complexity, while still making an accurate estimation. We can try to make a first-order approximation around the MPP with a formula in the form of $y = a(x - b) + c$. The model as given in [71] describes the current-voltage relation as given in Equation A.1.

$$v = V_T \ln\left[\frac{i}{I_{ph}} + \left(1 - \frac{i}{I_{ph}}\right) e^{\frac{V_{oc}}{V_T}}\right] - R_s i \tag{A.1}$$

To get the gradient of the I-V curve we differentiate this formula resulting in Equation A.2.

$$\frac{dI}{dV} = \frac{V_t}{\frac{I_{mpp}}{I_{ph}} + (1 - \frac{I_{mpp}}{I_{ph}}) e^{\frac{V_{oc}}{V_t n_s}}} \left(\frac{1 + e^{\frac{V_{oc}}{V_t n_s}}}{I_{ph}}\right) - R_s = a \tag{A.2}$$

Next $b = V_{mpp}$ and $c = I_{mpp}$. However, as I_{mpp} and V_{mpp} vary with G and T, a, b, and c are dependent on G and T.

Linearisation 2 G can be calculated according to Equation 2.9. This equation can be linearised by assuming the relation in Equation A.3:

$$\frac{G_0 i}{I_{sc}[1 + \alpha(T - T_0)]} >> G_0 e^{\frac{v + R_s i - V_{oc} - \beta(T - T_0)}{A V_T n_s}} \tag{A.3}$$

in which case $G \approx \frac{G_0 i}{I_{sc}[1+\alpha(T-T_0)]}$.

In this case, the gradient of the current when the illuminance changes is $\frac{di}{dG} = \frac{G_0}{i}$. Assuming the system works at its MPP, the equation becomes $\frac{di}{dG} = \frac{G_0}{I_{mpp,0}}$, in which $I_{mpp,0}$ is the MPP current at STC. This gradient can be calculated by using datasheet parameters only. This linearisation then gives an estimate of G by measuring only the input current according to Equation A.4.

$$G = \frac{I_{in} \times G_0}{I_{mpp,0}} \tag{A.4}$$

When measuring the short circuit current of a solar cell this is a very good approximation. However, the further away from that point you go, the more the influence of the exponential part becomes, and the worse this estimation becomes.

A linearisation thus can reduce the computational complexity but will introduce another error.

B.
Verilog code of LUT based design for on chip

B.1 Top level testbench

```
module top_level_tb();

  reg clk, reset, set_C, MPPT_search_done;
  reg [1:0] CP_bits;
  reg [2:0] CM_bits;
  reg [7:0] C_shift_set, ADC_out;
  reg [10:0] C_set;
  reg [10:0] reference;
  wire [10:0] result;
  wire VDD, VSS, done;

  top_level tl(VDD, VSS, clk, reset, reference, set_C, C_set, ADC_out, CP_bits, CM_bits,
  ↪  MPPT_search_done, result, done);

  initial begin
  reference = 313;
    ADC_out = 127;
    set_C = 0;
    C_set = 0;
    clk = 0;
    reset = 1;
    CP_bits = 1;
    CM_bits = 2;
    MPPT_search_done = 1;
  end

  always #10 clk = ~clk;

  initial #30 reset = 0;

endmodule
```

B.2 Top level

```
module top_level(
  inout VDD,
  inout VSS,
  input clk,
  input reset,
  input [10:0] reference,
  input set_C,
  input [10:0] C_set,
  input [7:0] ADC_out,
  input [1:0] CP_bits,
  input [2:0] CM_bits,
  input MPPT_search_done,
  output [10:0] result,
  output done,
  output [4:0] C_int,
  output [5:0] C_float
);

  wire [4:0] ADC_int, CP_int, Reff_int;
  wire [5:0] ADC_float, CP_float, Reff_float;
  wire [7:0] calib_shift, C_shift, shift_init;

  ill_calc_FSM calc_ill(VDD, VSS, clk, reset, ADC_int, ADC_float, CP_bits, Reff_int,
  ↪  Reff_float, reference, set_C, MPPT_search_done, done, result, C_int, C_float);
  LUT lut(VDD, VSS, CM_bits, Reff_int, Reff_float);
  ADC_LUT adc(VDD, VSS, ADC_out, ADC_int, ADC_float);

endmodule
```

B.3 Illuminance calculation FSM

```
module ill_calc_FSM(
  inout VDD,
  inout VSS,
  input clk,
  input reset,
  input [4:0] ADC_int,
  input [5:0] ADC_float,
  input [1:0] CP_int,
  //input [5:0] CP_float,
  input [4:0] Reff_int,
  input [5:0] Reff_float,
  input [10:0] reference,
  input C_set,
  input MPPT_search_done,
  output reg done,
  output reg [10:0] result,
  output reg [4:0] C_int,
  output reg [5:0] C_float
  );

  reg calibrated;
  reg [2:0] state;

  always @(posedge clk) begin
    if(reset) begin
      state <= 3'b000;
      done <= 0;
      calibrated <= 0;
      C_int <= 0;
      C_float <= 0;
    end
    case(state)
      3'b000  : begin
        done <= 0;
        state <= 3'b000;
        if(MPPT_search_done) begin
          if(calibrated)
            state <= 3'b010;
          else
            state <= 3'b001;
        end
      end
      3'b001  : begin
        done <= 0;
        state <= 3'b010;
        // C_set forces an external calibration value
        if(C_set) begin
          {C_int, C_float} <= reference;
          calibrated <= 1;
        end
      end
      3'b010  : begin
        done <= 0;
        state <= 3'b011;
```

```
        result <= {C_int, C_float} + {CP_int, 6'b000000} - {Reff_int, Reff_float} + {ADC_int,
        ↪  ADC_float};
        if(calibrated)
          state <= 3'b100;
      end
      // Calibration state, to calibrate set the reference, then reset this system
      3'b011  : begin
        done <= 0;
        {C_int, C_float} <= reference - result;
        calibrated <= 1;
        state <= 3'b010;
      end
      // Calculation is done
      3'b100  : begin
        done <= 1;
        state <= 3'b100;
        if(~MPPT_search_done)
          state <= 3'b000;
      end
      default : begin
        state <= 3'b000;
        done <= 0;
      end
    endcase
  end

endmodule
```

B.4 LUT

```
module LUT(
  inout VDD,
  inout VSS,
  input [3:0] CM_bits,
  output reg [4:0] Reff_int,
  output reg [5:0] Reff_float
  // input [1:0] CP_bits,
  // output reg [2:0] CP_int,
  // output reg [5:0] CP_float
  );

  /*
  always @(*) begin
      case(CP_bits)
        2'b00 : begin
          CP_int <= 1;
          // CP_float <= 6'b000000;
        end
        2'b01 : begin
          CP_int <= 2;
          // CP_float <= 6'b010101;
        end
        2'b10 : begin
          CP_int <= 3;
          // CP_float <= 6'b000000;
        end
        2'b11 : begin
          CP_int <= 4;
          // CP_float <= 6'b000000;
        end
        default begin
          CP_int <= 1;
          // CP_float <= 6'b000000;
        end
      endcase
    end
    */

    // 1 has been added because of the CP translation
    always @(*) begin
      case(CM_bits)
        4'b0001  : begin
          Reff_int <= 16;
          Reff_float <= 6'b100111;
        end
        4'b0011  : begin
          Reff_int <= 12;
          Reff_float <= 6'b100111;
        end
        4'b0111  : begin
          Reff_int <= 8;
          Reff_float <= 6'b100111;
        end
        4'b1111  : begin
```

```
          Reff_int <= 4;
          Reff_float <= 6'b100111;
        end
        default : begin
          Reff_int <= 16;
          Reff_float <= 6'b100111;
        end
      endcase
    end

endmodule
```

B.5 ADC LUT

```
 module ADC_LUT(
  inout VDD,
  inout VSS,
  input [7:0] ADC_bits,
  output reg [4:0] ADC_int,
  output reg [5:0] ADC_float
  );


 always @(*) begin
 case(ADC_bits)

 8'b00000000  : begin
 ADC_int <=  0 ;
 ADC_float <=  0 ;
end
8'b00000001  : begin
 ADC_int <=  0 ;
 ADC_float <=  0 ;
end
8'b00000010  : begin
 ADC_int <=  1 ;
 ADC_float <=  0 ;
end
8'b00000011  : begin
 ADC_int <=  1 ;
 ADC_float <=  37 ;
end
8'b00000100  : begin
 ADC_int <=  2 ;
 ADC_float <=  0 ;
end
8'b00000101  : begin
 ADC_int <=  2 ;
 ADC_float <=  21 ;
end
8'b00000110  : begin
 ADC_int <=  2 ;
 ADC_float <=  37 ;
end
8'b00000111  : begin
 ADC_int <=  2 ;
 ADC_float <=  52 ;
end
8'b00001000  : begin
 ADC_int <=  3 ;
 ADC_float <=  0 ;
end
8'b00001001  : begin
 ADC_int <=  3 ;
 ADC_float <=  11 ;
end
8'b00001010  : begin
 ADC_int <=  3 ;
```

```
 ADC_float <=  21 ;
end
8'b00001011  : begin
 ADC_int <=  3 ;
 ADC_float <=  29 ;
end
8'b00001100  : begin
 ADC_int <=  3 ;
 ADC_float <=  37 ;
end
8'b00001101  : begin
 ADC_int <=  3 ;
 ADC_float <=  45 ;
end
8'b00001110  : begin
 ADC_int <=  3 ;
 ADC_float <=  52 ;
end
8'b00001111  : begin
 ADC_int <=  3 ;
 ADC_float <=  58 ;
end
8'b00010000  : begin
 ADC_int <=  4 ;
 ADC_float <=  0 ;
end
8'b00010001  : begin
 ADC_int <=  4 ;
 ADC_float <=  6 ;
end
8'b00010010  : begin
 ADC_int <=  4 ;
 ADC_float <=  11 ;
end
8'b00010011  : begin
 ADC_int <=  4 ;
 ADC_float <=  16 ;
end
8'b00010100  : begin
 ADC_int <=  4 ;
 ADC_float <=  21 ;
end
8'b00010101  : begin
 ADC_int <=  4 ;
 ADC_float <=  25 ;
end
8'b00010110  : begin
 ADC_int <=  4 ;
 ADC_float <=  29 ;
end
8'b00010111  : begin
 ADC_int <=  4 ;
 ADC_float <=  34 ;
end
8'b00011000  : begin
 ADC_int <=  4 ;
```

```
 ADC_float <=  37 ;
end
8'b00011001  : begin
 ADC_int <=  4 ;
 ADC_float <=  41 ;
end
8'b00011010  : begin
 ADC_int <=  4 ;
 ADC_float <=  45 ;
end
8'b00011011  : begin
 ADC_int <=  4 ;
 ADC_float <=  48 ;
end
8'b00011100  : begin
 ADC_int <=  4 ;
 ADC_float <=  52 ;
end
8'b00011101  : begin
 ADC_int <=  4 ;
 ADC_float <=  55 ;
end
8'b00011110  : begin
 ADC_int <=  4 ;
 ADC_float <=  58 ;
end
8'b00011111  : begin
 ADC_int <=  4 ;
 ADC_float <=  61 ;
end
8'b00100000  : begin
 ADC_int <=  5 ;
 ADC_float <=  0 ;
end
8'b00100001  : begin
 ADC_int <=  5 ;
 ADC_float <=  3 ;
end
8'b00100010  : begin
 ADC_int <=  5 ;
 ADC_float <=  6 ;
end
8'b00100011  : begin
 ADC_int <=  5 ;
 ADC_float <=  8 ;
end
8'b00100100  : begin
 ADC_int <=  5 ;
 ADC_float <=  11 ;
end
8'b00100101  : begin
 ADC_int <=  5 ;
 ADC_float <=  13 ;
end
8'b00100110  : begin
 ADC_int <=  5 ;
```

```
 ADC_float <=  16 ;
end
8'b00100111  : begin
 ADC_int <=  5 ;
 ADC_float <=  18 ;
end
8'b00101000  : begin
 ADC_int <=  5 ;
 ADC_float <=  21 ;
end
8'b00101001  : begin
 ADC_int <=  5 ;
 ADC_float <=  23 ;
end
8'b00101010  : begin
 ADC_int <=  5 ;
 ADC_float <=  25 ;
end
8'b00101011  : begin
 ADC_int <=  5 ;
 ADC_float <=  27 ;
end
8'b00101100  : begin
 ADC_int <=  5 ;
 ADC_float <=  29 ;
end
8'b00101101  : begin
 ADC_int <=  5 ;
 ADC_float <=  31 ;
end
8'b00101110  : begin
 ADC_int <=  5 ;
 ADC_float <=  34 ;
end
8'b00101111  : begin
 ADC_int <=  5 ;
 ADC_float <=  35 ;
end
8'b00110000  : begin
 ADC_int <=  5 ;
 ADC_float <=  37 ;
end
8'b00110001  : begin
 ADC_int <=  5 ;
 ADC_float <=  39 ;
end
8'b00110010  : begin
 ADC_int <=  5 ;
 ADC_float <=  41 ;
end
8'b00110011  : begin
 ADC_int <=  5 ;
 ADC_float <=  43 ;
end
8'b00110100  : begin
 ADC_int <=  5 ;
```

```
 ADC_float <=  45 ;
end
8'b00110101  : begin
 ADC_int <=  5 ;
 ADC_float <=  47 ;
end
8'b00110110  : begin
 ADC_int <=  5 ;
 ADC_float <=  48 ;
end
8'b00110111  : begin
 ADC_int <=  5 ;
 ADC_float <=  50 ;
end
8'b00111000  : begin
 ADC_int <=  5 ;
 ADC_float <=  52 ;
end
8'b00111001  : begin
 ADC_int <=  5 ;
 ADC_float <=  53 ;
end
8'b00111010  : begin
 ADC_int <=  5 ;
 ADC_float <=  55 ;
end
8'b00111011  : begin
 ADC_int <=  5 ;
 ADC_float <=  56 ;
end
8'b00111100  : begin
 ADC_int <=  5 ;
 ADC_float <=  58 ;
end
8'b00111101  : begin
 ADC_int <=  5 ;
 ADC_float <=  60 ;
end
8'b00111110  : begin
 ADC_int <=  5 ;
 ADC_float <=  61 ;
end
8'b00111111  : begin
 ADC_int <=  5 ;
 ADC_float <=  63 ;
end
8'b01000000  : begin
 ADC_int <=  6 ;
 ADC_float <=  0 ;
end
8'b01000001  : begin
 ADC_int <=  6 ;
 ADC_float <=  1 ;
end
8'b01000010  : begin
 ADC_int <=  6 ;
```

```
 ADC_float <=  3 ;
end
8'b01000011  : begin
 ADC_int <=  6 ;
 ADC_float <=  4 ;
end
8'b01000100  : begin
 ADC_int <=  6 ;
 ADC_float <=  6 ;
end
8'b01000101  : begin
 ADC_int <=  6 ;
 ADC_float <=  7 ;
end
8'b01000110  : begin
 ADC_int <=  6 ;
 ADC_float <=  8 ;
end
8'b01000111  : begin
 ADC_int <=  6 ;
 ADC_float <=  10 ;
end
8'b01001000  : begin
 ADC_int <=  6 ;
 ADC_float <=  11 ;
end
8'b01001001  : begin
 ADC_int <=  6 ;
 ADC_float <=  12 ;
end
8'b01001010  : begin
 ADC_int <=  6 ;
 ADC_float <=  13 ;
end
8'b01001011  : begin
 ADC_int <=  6 ;
 ADC_float <=  15 ;
end
8'b01001100  : begin
 ADC_int <=  6 ;
 ADC_float <=  16 ;
end
8'b01001101  : begin
 ADC_int <=  6 ;
 ADC_float <=  17 ;
end
8'b01001110  : begin
 ADC_int <=  6 ;
 ADC_float <=  18 ;
end
8'b01001111  : begin
 ADC_int <=  6 ;
 ADC_float <=  19 ;
end
8'b01010000  : begin
 ADC_int <=  6 ;
```

```
 ADC_float <=  21 ;
end
8'b01010001  : begin
 ADC_int <=  6 ;
 ADC_float <=  22 ;
end
8'b01010010  : begin
 ADC_int <=  6 ;
 ADC_float <=  23 ;
end
8'b01010011  : begin
 ADC_int <=  6 ;
 ADC_float <=  24 ;
end
8'b01010100  : begin
 ADC_int <=  6 ;
 ADC_float <=  25 ;
end
8'b01010101  : begin
 ADC_int <=  6 ;
 ADC_float <=  26 ;
end
8'b01010110  : begin
 ADC_int <=  6 ;
 ADC_float <=  27 ;
end
8'b01010111  : begin
 ADC_int <=  6 ;
 ADC_float <=  28 ;
end
8'b01011000  : begin
 ADC_int <=  6 ;
 ADC_float <=  29 ;
end
8'b01011001  : begin
 ADC_int <=  6 ;
 ADC_float <=  30 ;
end
8'b01011010  : begin
 ADC_int <=  6 ;
 ADC_float <=  31 ;
end
8'b01011011  : begin
 ADC_int <=  6 ;
 ADC_float <=  32 ;
end
8'b01011100  : begin
 ADC_int <=  6 ;
 ADC_float <=  34 ;
end
8'b01011101  : begin
 ADC_int <=  6 ;
 ADC_float <=  35 ;
end
8'b01011110  : begin
 ADC_int <=  6 ;
```

```
 ADC_float <=  35 ;
end
8'b01011111  : begin
 ADC_int <=  6 ;
 ADC_float <=  36 ;
end
8'b01100000  : begin
 ADC_int <=  6 ;
 ADC_float <=  37 ;
end
8'b01100001  : begin
 ADC_int <=  6 ;
 ADC_float <=  38 ;
end
8'b01100010  : begin
 ADC_int <=  6 ;
 ADC_float <=  39 ;
end
8'b01100011  : begin
 ADC_int <=  6 ;
 ADC_float <=  40 ;
end
8'b01100100  : begin
 ADC_int <=  6 ;
 ADC_float <=  41 ;
end
8'b01100101  : begin
 ADC_int <=  6 ;
 ADC_float <=  42 ;
end
8'b01100110  : begin
 ADC_int <=  6 ;
 ADC_float <=  43 ;
end
8'b01100111  : begin
 ADC_int <=  6 ;
 ADC_float <=  44 ;
end
8'b01101000  : begin
 ADC_int <=  6 ;
 ADC_float <=  45 ;
end
8'b01101001  : begin
 ADC_int <=  6 ;
 ADC_float <=  46 ;
end
8'b01101010  : begin
 ADC_int <=  6 ;
 ADC_float <=  47 ;
end
8'b01101011  : begin
 ADC_int <=  6 ;
 ADC_float <=  47 ;
end
8'b01101100  : begin
 ADC_int <=  6 ;
```

```
 ADC_float <=  48 ;
end
8'b01101101  : begin
 ADC_int <=  6 ;
 ADC_float <=  49 ;
end
8'b01101110  : begin
 ADC_int <=  6 ;
 ADC_float <=  50 ;
end
8'b01101111  : begin
 ADC_int <=  6 ;
 ADC_float <=  51 ;
end
8'b01110000  : begin
 ADC_int <=  6 ;
 ADC_float <=  52 ;
end
8'b01110001  : begin
 ADC_int <=  6 ;
 ADC_float <=  52 ;
end
8'b01110010  : begin
 ADC_int <=  6 ;
 ADC_float <=  53 ;
end
8'b01110011  : begin
 ADC_int <=  6 ;
 ADC_float <=  54 ;
end
8'b01110100  : begin
 ADC_int <=  6 ;
 ADC_float <=  55 ;
end
8'b01110101  : begin
 ADC_int <=  6 ;
 ADC_float <=  56 ;
end
8'b01110110  : begin
 ADC_int <=  6 ;
 ADC_float <=  56 ;
end
8'b01110111  : begin
 ADC_int <=  6 ;
 ADC_float <=  57 ;
end
8'b01111000  : begin
 ADC_int <=  6 ;
 ADC_float <=  58 ;
end
8'b01111001  : begin
 ADC_int <=  6 ;
 ADC_float <=  59 ;
end
8'b01111010  : begin
 ADC_int <=  6 ;
```

```
 ADC_float <=  60 ;
end
8'b01111011  : begin
 ADC_int <=  6 ;
 ADC_float <=  60 ;
end
8'b01111100  : begin
 ADC_int <=  6 ;
 ADC_float <=  61 ;
end
8'b01111101  : begin
 ADC_int <=  6 ;
 ADC_float <=  62 ;
end
8'b01111110  : begin
 ADC_int <=  6 ;
 ADC_float <=  63 ;
end
8'b01111111  : begin
 ADC_int <=  6 ;
 ADC_float <=  63 ;
end
8'b10000000  : begin
 ADC_int <=  7 ;
 ADC_float <=  0 ;
end
8'b10000001  : begin
 ADC_int <=  7 ;
 ADC_float <=  1 ;
end
8'b10000010  : begin
 ADC_int <=  7 ;
 ADC_float <=  1 ;
end
8'b10000011  : begin
 ADC_int <=  7 ;
 ADC_float <=  2 ;
end
8'b10000100  : begin
 ADC_int <=  7 ;
 ADC_float <=  3 ;
end
8'b10000101  : begin
 ADC_int <=  7 ;
 ADC_float <=  4 ;
end
8'b10000110  : begin
 ADC_int <=  7 ;
 ADC_float <=  4 ;
end
8'b10000111  : begin
 ADC_int <=  7 ;
 ADC_float <=  5 ;
end
8'b10001000  : begin
 ADC_int <=  7 ;
```

```
 ADC_float <=  6 ;
end
8'b10001001  : begin
 ADC_int <=  7 ;
 ADC_float <=  6 ;
end
8'b10001010  : begin
 ADC_int <=  7 ;
 ADC_float <=  7 ;
end
8'b10001011  : begin
 ADC_int <=  7 ;
 ADC_float <=  8 ;
end
8'b10001100  : begin
 ADC_int <=  7 ;
 ADC_float <=  8 ;
end
8'b10001101  : begin
 ADC_int <=  7 ;
 ADC_float <=  9 ;
end
8'b10001110  : begin
 ADC_int <=  7 ;
 ADC_float <=  10 ;
end
8'b10001111  : begin
 ADC_int <=  7 ;
 ADC_float <=  10 ;
end
8'b10010000  : begin
 ADC_int <=  7 ;
 ADC_float <=  11 ;
end
8'b10010001  : begin
 ADC_int <=  7 ;
 ADC_float <=  12 ;
end
8'b10010010  : begin
 ADC_int <=  7 ;
 ADC_float <=  12 ;
end
8'b10010011  : begin
 ADC_int <=  7 ;
 ADC_float <=  13 ;
end
8'b10010100  : begin
 ADC_int <=  7 ;
 ADC_float <=  13 ;
end
8'b10010101  : begin
 ADC_int <=  7 ;
 ADC_float <=  14 ;
end
8'b10010110  : begin
 ADC_int <=  7 ;
```

```
 ADC_float <=  15 ;
end
8'b10010111  : begin
 ADC_int <=  7 ;
 ADC_float <=  15 ;
end
8'b10011000  : begin
 ADC_int <=  7 ;
 ADC_float <=  16 ;
end
8'b10011001  : begin
 ADC_int <=  7 ;
 ADC_float <=  16 ;
end
8'b10011010  : begin
 ADC_int <=  7 ;
 ADC_float <=  17 ;
end
8'b10011011  : begin
 ADC_int <=  7 ;
 ADC_float <=  18 ;
end
8'b10011100  : begin
 ADC_int <=  7 ;
 ADC_float <=  18 ;
end
8'b10011101  : begin
 ADC_int <=  7 ;
 ADC_float <=  19 ;
end
8'b10011110  : begin
 ADC_int <=  7 ;
 ADC_float <=  19 ;
end
8'b10011111  : begin
 ADC_int <=  7 ;
 ADC_float <=  20 ;
end
8'b10100000  : begin
 ADC_int <=  7 ;
 ADC_float <=  21 ;
end
8'b10100001  : begin
 ADC_int <=  7 ;
 ADC_float <=  21 ;
end
8'b10100010  : begin
 ADC_int <=  7 ;
 ADC_float <=  22 ;
end
8'b10100011  : begin
 ADC_int <=  7 ;
 ADC_float <=  22 ;
end
8'b10100100  : begin
 ADC_int <=  7 ;
```

```
 ADC_float <=  23 ;
end
8'b10100101  : begin
 ADC_int <=  7 ;
 ADC_float <=  23 ;
end
8'b10100110  : begin
 ADC_int <=  7 ;
 ADC_float <=  24 ;
end
8'b10100111  : begin
 ADC_int <=  7 ;
 ADC_float <=  25 ;
end
8'b10101000  : begin
 ADC_int <=  7 ;
 ADC_float <=  25 ;
end
8'b10101001  : begin
 ADC_int <=  7 ;
 ADC_float <=  26 ;
end
8'b10101010  : begin
 ADC_int <=  7 ;
 ADC_float <=  26 ;
end
8'b10101011  : begin
 ADC_int <=  7 ;
 ADC_float <=  27 ;
end
8'b10101100  : begin
 ADC_int <=  7 ;
 ADC_float <=  27 ;
end
8'b10101101  : begin
 ADC_int <=  7 ;
 ADC_float <=  28 ;
end
8'b10101110  : begin
 ADC_int <=  7 ;
 ADC_float <=  28 ;
end
8'b10101111  : begin
 ADC_int <=  7 ;
 ADC_float <=  29 ;
end
8'b10110000  : begin
 ADC_int <=  7 ;
 ADC_float <=  29 ;
end
8'b10110001  : begin
 ADC_int <=  7 ;
 ADC_float <=  30 ;
end
8'b10110010  : begin
 ADC_int <=  7 ;
```

```
 ADC_float <=  30 ;
end
8'b10110011  : begin
 ADC_int <=  7 ;
 ADC_float <=  31 ;
end
8'b10110100  : begin
 ADC_int <=  7 ;
 ADC_float <=  31 ;
end
8'b10110101  : begin
 ADC_int <=  7 ;
 ADC_float <=  32 ;
end
8'b10110110  : begin
 ADC_int <=  7 ;
 ADC_float <=  32 ;
end
8'b10110111  : begin
 ADC_int <=  7 ;
 ADC_float <=  33 ;
end
8'b10111000  : begin
 ADC_int <=  7 ;
 ADC_float <=  34 ;
end
8'b10111001  : begin
 ADC_int <=  7 ;
 ADC_float <=  34 ;
end
8'b10111010  : begin
 ADC_int <=  7 ;
 ADC_float <=  35 ;
end
8'b10111011  : begin
 ADC_int <=  7 ;
 ADC_float <=  35 ;
end
8'b10111100  : begin
 ADC_int <=  7 ;
 ADC_float <=  35 ;
end
8'b10111101  : begin
 ADC_int <=  7 ;
 ADC_float <=  36 ;
end
8'b10111110  : begin
 ADC_int <=  7 ;
 ADC_float <=  36 ;
end
8'b10111111  : begin
 ADC_int <=  7 ;
 ADC_float <=  37 ;
end
8'b11000000  : begin
 ADC_int <=  7 ;
```

```
 ADC_float <=  37 ;
end
8'b11000001  : begin
 ADC_int <=  7 ;
 ADC_float <=  38 ;
end
8'b11000010  : begin
 ADC_int <=  7 ;
 ADC_float <=  38 ;
end
8'b11000011  : begin
 ADC_int <=  7 ;
 ADC_float <=  39 ;
end
8'b11000100  : begin
 ADC_int <=  7 ;
 ADC_float <=  39 ;
end
8'b11000101  : begin
 ADC_int <=  7 ;
 ADC_float <=  40 ;
end
8'b11000110  : begin
 ADC_int <=  7 ;
 ADC_float <=  40 ;
end
8'b11000111  : begin
 ADC_int <=  7 ;
 ADC_float <=  41 ;
end
8'b11001000  : begin
 ADC_int <=  7 ;
 ADC_float <=  41 ;
end
8'b11001001  : begin
 ADC_int <=  7 ;
 ADC_float <=  42 ;
end
8'b11001010  : begin
 ADC_int <=  7 ;
 ADC_float <=  42 ;
end
8'b11001011  : begin
 ADC_int <=  7 ;
 ADC_float <=  43 ;
end
8'b11001100  : begin
 ADC_int <=  7 ;
 ADC_float <=  43 ;
end
8'b11001101  : begin
 ADC_int <=  7 ;
 ADC_float <=  43 ;
end
8'b11001110  : begin
 ADC_int <=  7 ;
```

```
 ADC_float <=  44 ;
end
8'b11001111  : begin
 ADC_int <=  7 ;
 ADC_float <=  44 ;
end
8'b11010000  : begin
 ADC_int <=  7 ;
 ADC_float <=  45 ;
end
8'b11010001  : begin
 ADC_int <=  7 ;
 ADC_float <=  45 ;
end
8'b11010010  : begin
 ADC_int <=  7 ;
 ADC_float <=  46 ;
end
8'b11010011  : begin
 ADC_int <=  7 ;
 ADC_float <=  46 ;
end
8'b11010100  : begin
 ADC_int <=  7 ;
 ADC_float <=  47 ;
end
8'b11010101  : begin
 ADC_int <=  7 ;
 ADC_float <=  47 ;
end
8'b11010110  : begin
 ADC_int <=  7 ;
 ADC_float <=  47 ;
end
8'b11010111  : begin
 ADC_int <=  7 ;
 ADC_float <=  48 ;
end
8'b11011000  : begin
 ADC_int <=  7 ;
 ADC_float <=  48 ;
end
8'b11011001  : begin
 ADC_int <=  7 ;
 ADC_float <=  49 ;
end
8'b11011010  : begin
 ADC_int <=  7 ;
 ADC_float <=  49 ;
end
8'b11011011  : begin
 ADC_int <=  7 ;
 ADC_float <=  50 ;
end
8'b11011100  : begin
 ADC_int <=  7 ;
```

```
 ADC_float <=  50 ;
end
8'b11011101  : begin
 ADC_int <=  7 ;
 ADC_float <=  50 ;
end
8'b11011110  : begin
 ADC_int <=  7 ;
 ADC_float <=  51 ;
end
8'b11011111  : begin
 ADC_int <=  7 ;
 ADC_float <=  51 ;
end
8'b11100000  : begin
 ADC_int <=  7 ;
 ADC_float <=  52 ;
end
8'b11100001  : begin
 ADC_int <=  7 ;
 ADC_float <=  52 ;
end
8'b11100010  : begin
 ADC_int <=  7 ;
 ADC_float <=  52 ;
end
8'b11100011  : begin
 ADC_int <=  7 ;
 ADC_float <=  53 ;
end
8'b11100100  : begin
 ADC_int <=  7 ;
 ADC_float <=  53 ;
end
8'b11100101  : begin
 ADC_int <=  7 ;
 ADC_float <=  54 ;
end
8'b11100110  : begin
 ADC_int <=  7 ;
 ADC_float <=  54 ;
end
8'b11100111  : begin
 ADC_int <=  7 ;
 ADC_float <=  55 ;
end
8'b11101000  : begin
 ADC_int <=  7 ;
 ADC_float <=  55 ;
end
8'b11101001  : begin
 ADC_int <=  7 ;
 ADC_float <=  55 ;
end
8'b11101010  : begin
 ADC_int <=  7 ;
```

```
 ADC_float <=  56 ;
end
8'b11101011  : begin
 ADC_int <=  7 ;
 ADC_float <=  56 ;
end
8'b11101100  : begin
 ADC_int <=  7 ;
 ADC_float <=  56 ;
end
8'b11101101  : begin
 ADC_int <=  7 ;
 ADC_float <=  57 ;
end
8'b11101110  : begin
 ADC_int <=  7 ;
 ADC_float <=  57 ;
end
8'b11101111  : begin
 ADC_int <=  7 ;
 ADC_float <=  58 ;
end
8'b11110000  : begin
 ADC_int <=  7 ;
 ADC_float <=  58 ;
end
8'b11110001  : begin
 ADC_int <=  7 ;
 ADC_float <=  58 ;
end
8'b11110010  : begin
 ADC_int <=  7 ;
 ADC_float <=  59 ;
end
8'b11110011  : begin
 ADC_int <=  7 ;
 ADC_float <=  59 ;
end
8'b11110100  : begin
 ADC_int <=  7 ;
 ADC_float <=  60 ;
end
8'b11110101  : begin
 ADC_int <=  7 ;
 ADC_float <=  60 ;
end
8'b11110110  : begin
 ADC_int <=  7 ;
 ADC_float <=  60 ;
end
8'b11110111  : begin
 ADC_int <=  7 ;
 ADC_float <=  61 ;
end
8'b11111000  : begin
 ADC_int <=  7 ;
```

```
 ADC_float <=  61 ;
end
8'b11111001  : begin
 ADC_int <=  7 ;
 ADC_float <=  61 ;
end
8'b11111010  : begin
 ADC_int <=  7 ;
 ADC_float <=  62 ;
end
8'b11111011  : begin
 ADC_int <=  7 ;
 ADC_float <=  62 ;
end
8'b11111100  : begin
 ADC_int <=  7 ;
 ADC_float <=  63 ;
end
8'b11111101  : begin
 ADC_int <=  7 ;
 ADC_float <=  63 ;
end
8'b11111110  : begin
 ADC_int <=  7 ;
 ADC_float <=  63 ;
end
8'b11111111  : begin
 ADC_int <=  7 ;
 ADC_float <=  63 ;
end
default : begin
  ADC_int <= 0;
  ADC_float <= 0;
end
endcase
end
endmodule
```

C.
Verilog code of algorithm based design for on chip

C.1 Top level testbench

```
module top_level_tb();

  reg clk, reset, set_C;
  reg [1:0] CP_bits,
  reg [2:0] CM_bits;
  reg [7:0] C_shift_set, ADC_out;
  reg [10:0] C_set;
  reg [10:0] reference;
  wire [10:0] result;
  wire VDD, VSS, done;

  top_level tl(VDD, VSS, clk, reset, reference, set_C, C_set, ADC_out, CP_bits, CM_bits,
  ↪  result, done);

  initial begin
    reference = 212;
    ADC_out = 145;
    set_C = 0;
    C_set = 0;
    clk = 0;
    reset = 1;
    CP_bits = 3;
    CM_bits = 1;
  end

  always #10 clk = ~clk;

  initial #30 reset = 0;

  pullup(sda);
  pullup(scl);

endmodule
```

C.2 Top level

```
module top_level(
  inout VDD,
  inout VSS,
  input clk,
  input reset,
  input [10:0] reference,
  input set_C,
  input [10:0] C_set,
  input [7:0] ADC_out,
  input [1:0] CP_bits,
  input [2:0] CM_bits,
  output [10:0] result,
  output done
);

  wire [4:0] CP_int, Reff_int;
  wire [5:0] CP_float, Reff_float;
  wire [7:0] calib_shift, C_shift, shift_init;

  ill_calc_FSM calc_ill(VDD, VSS, clk, reset, ADC_out, CP_bits, Reff_int, Reff_float,
  ↪  reference, set_C, done, result);
  LUT lut(VDD, VSS, CM_bits, Reff_int, Reff_float);

endmodule
```

C.3 Illuminance calculation FSM

```
module ill_calc_FSM(
  inout VDD,
  inout VSS,
  input clk,
  input reset,
  input [7:0] ADC_out,
  input [4:0] CP_int,
  input [5:0] CP_float,
  input [4:0] Reff_int,
  input [5:0] Reff_float,
  input [10:0] reference,
  input C_set,
  output reg done,
  output reg [10:0] result
  );


  wire dlc_done;
  wire [6:0] log_int;
  wire [5:0] log_float;

  reg dlc_rst, calibrated;
  reg [2:0] state;
  reg [5:0] C_int;
  reg [5:0] C_float;

  dec_log_conv dlc(VDD, VSS, clk, dlc_rst, ADC_out, log_int, log_float, dlc_done);


  always @(posedge clk) begin
    if(reset) begin
      state <= 3'b000;
      done <= 0;
      dlc_rst <= 1;
      calibrated <= 0;
      C_int <= 0;
      C_float <= 0;
    end
    case(state)
      3'b000  : begin
        done <= 0;
        dlc_rst <= 1;
        if(calibrated)
          state <= 3'b001;
        else
          state <= 3'b111;
      end
      3'b111  : begin
        done <= 0;
        dlc_rst <= 1;
        state <= 3'b001;
        if(C_set) begin
          state <= 3'b110;
          {C_int, C_float} <= reference;
```

```
          calibrated <= 1;
        end
      end
      3'b001  : begin
        done <= 0;
        dlc_rst <= 0;
        if(dlc_done)
          state <= 3'b010;
      end
      3'b010  : begin
        done <= 0;
        dlc_rst <= 0;
        state <= 3'b011;
        dlc_rst <= 1;
        result <= {C_int, C_float} + {CP_int, CP_float} - {Reff_int, Reff_float} + {log_int,
        ↪  log_float};
        if(calibrated)
          state <= 3'b100;
      end
      3'b011  : begin
        done <= 0;
        dlc_rst <= 1;
        {C_int, C_float} <= reference - result;
        calibrated <= 1;
        state <= 3'b000;
      end
      3'b100  : begin
        dlc_rst <= 1;
        done <= 1;
        state <= 3'b000;
      end
      default : begin
        dlc_rst <= 1;
        state <= 3'b000;
        done <= 0;
      end
    endcase
  end

endmodule
```

C.4 LUT

```
module LUT(
  inout VDD,
  inout VSS,
  input [3:0] CM_bits,
  output reg [4:0] Reff_int,
  output reg [5:0] Reff_float
  // input [1:0] CP_bits,
  // output reg [2:0] CP_int,
  // output reg [5:0] CP_float
  );

  /*
  always @(*) begin
      case(CP_bits)
        2'b00 : begin
          CP_int <= 1;
          // CP_float <= 6'b000000;
        end
        2'b01 : begin
          CP_int <= 2;
          // CP_float <= 6'b010101;
        end
        2'b10 : begin
          CP_int <= 3;
          // CP_float <= 6'b000000;
        end
        2'b11 : begin
          CP_int <= 4;
          // CP_float <= 6'b000000;
        end
        default begin
          CP_int <= 1;
          // CP_float <= 6'b000000;
        end
      endcase
    end
    */

    // 1 has been added because of the CP translation
    always @(*) begin
      case(CM_bits)
        4'b0001  : begin
          Reff_int <= 16;
          Reff_float <= 6'b100111;
        end
        4'b0011  : begin
          Reff_int <= 12;
          Reff_float <= 6'b100111;
        end
        4'b0111  : begin
          Reff_int <= 8;
          Reff_float <= 6'b100111;
        end
        4'b1111  : begin
```

```
          Reff_int <= 4;
          Reff_float <= 6'b100111;
        end
        default : begin
          Reff_int <= 16;
          Reff_float <= 6'b100111;
        end
      endcase
    end

endmodule
```

C.5 Decimal to base-2 logarithm converter

```
module dec_log_conv(
  inout VDD,
  inout VSS,
  input clk,
  input reset,
  input [7:0] dec,
  output reg [6:0] log_int,
  output reg [5:0] log_float,
  output reg done
  );

  reg [2:0] count, next_count;
  reg [3:0] state, next_state;
  reg [6:0] log_int_next;
  reg [5:0] z, z_next;
  reg [7:0] m, m_next;
  reg [7:0] square_in;
  wire [15:0] square_out;

  square_comb square(VDD, VSS, square_in, square_out);


  always @(posedge clk) begin
    if(reset) begin
      state <= 4'b0000;
      z <= 0;
      m <= 0;
      count <= 0;
      log_int <= 0;
    end
    else begin
      state <= next_state;
      z <= z_next;
      m <= m_next;
      count <= next_count;
      log_int <= log_int_next;
    end
  end

  always @(posedge clk) begin
    if(reset) begin
      z_next <= 0;
      log_float <= 0;
      done <= 0;
      log_int_next <= 0;
      next_count <= 0;
      next_state <= 4'b0000;
    end
    else begin
      case(state)
        4'b0000  : begin
          z_next <= 0;
          m_next <= dec[7:0];
          done <= 0;
```

```
    log_int_next <= 6;
    next_count <= 0;
    next_state <= 4'b0001;
  end
  4'b0001  : begin
    if(m == 0)
      next_state <= 4'b1001;
    else if(m[7]) begin
      m_next <= m >> 1;
      log_int_next <= 7;
      next_state <= 4'b0011;
    end
    else if(m[6])
      next_state <= 4'b0011;
    else
      next_state <= 4'b0010;
  end
  4'b0010  : begin
    m_next <= m << 1;
    log_int_next <= log_int - 1;
    next_state <= 4'b0001;
  end
  4'b0011  : begin
    z_next <= z << 1;
    square_in <=  m[6:0];
    next_state <= 4'b0100;
  end
  4'b0100  : begin
    m_next <= square_out[13:6];
    next_state <= 4'b0101;
  end
  4'b0101  : begin
    if(m[7])
      next_state <= 4'b0110;
    else
      next_state <= 4'b0111;
  end
  4'b0110  : begin
    z_next <= z+1;
    m_next <= m >> 1;
    next_state <= 4'b0111;
  end
  4'b0111 : begin
    next_count <= count + 1;
    if(count == 5)
      next_state <= 4'b1000;
    else
      next_state <= 4'b0011;
  end
  4'b1000  : begin
    done <= 1;
    next_state <= 4'b0000;
    log_float <= z;
  end
  4'b1001          : begin
    done <= 1;
```

```
          log_float <= 0;
          log_int_next <= 0;
        end
      endcase
    end
  end

endmodule
```

C.6 Squaring unit

```
module square_comb(
  inout VDD,
  inout VSS,
  input [6:0] A,
  output [13:0] out
  );

  assign out = A*A;


endmodule
```

D.
Verilog and VHDL code for the FPGA implementation

D.1 Top level

```
module top_level(
  input clk,
  input reset,
  input set_C,
  input read_ref,
  inout sda,
  inout scl,
  output [10:0] result,
  output done,
  output [2:0] FB,              // For debugging purposes
  output ack_error              // For debugging purposes
);

  wire I2C_done;
  wire [1:0] CP_bits;
  wire [3:0] CM_bits;
  wire [4:0] ADC_int, Reff_int;
  wire [5:0] ADC_float, Reff_float;
  wire [7:0] calib_shift, C_shift, shift_init, ADC_out;
  wire [10:0] reference;

  ill_calc_FSM calc_ill(clk, reset, ADC_int, ADC_float, CP_bits, Reff_int, Reff_float,
  ↪  reference, set_C, I2C_done, done, result);
  LUT lut(CM_bits, Reff_int, Reff_float);
  ADC_LUT adc(ADC_out, ADC_int, ADC_float);
  I2CControl I2C(clk, reset, read_ref, result, sda, scl, ADC_out, CP_bits, CM_bits, I2C_done,
  ↪  FB, reference, ack_error);

endmodule
```

D.2 Illuminance calculation FSM

```
module ill_calc_FSM(
  inout VDD,
  inout VSS,
  input clk,
  input reset,
  input [4:0] ADC_int,
  input [5:0] ADC_float,
  input [1:0] CP_int,
  //input [5:0] CP_float,
  input [4:0] Reff_int,
  input [5:0] Reff_float,
  input [10:0] reference,
  input C_set,
  input MPPT_search_done,
  output reg done,
  output reg [10:0] result,
  output reg [4:0] C_int,
  output reg [5:0] C_float
  );

  reg calibrated;
  reg [2:0] state;

  always @(posedge clk) begin
    if(reset) begin
      state <= 3'b000;
      done <= 0;
      calibrated <= 0;
      C_int <= 0;
      C_float <= 0;
    end
    case(state)
      3'b000  : begin
        done <= 0;
        state <= 3'b000;
        if(MPPT_search_done) begin
          if(calibrated)
            state <= 3'b010;
          else
            state <= 3'b001;
        end
      end
      3'b001  : begin
        done <= 0;
        state <= 3'b010;
        // C_set forces an external calibration value
        if(C_set) begin
          {C_int, C_float} <= reference;
          calibrated <= 1;
        end
      end
      3'b010  : begin
        done <= 0;
        state <= 3'b011;
```

```
        result <= {C_int, C_float} + {CP_int, 6'b000000} - {Reff_int, Reff_float} + {ADC_int,
        ↪  ADC_float};
        if(calibrated)
          state <= 3'b100;
      end
      // Calibration state, to calibrate set the reference, then reset this system
      3'b011  : begin
        done <= 0;
        {C_int, C_float} <= reference - result;
        calibrated <= 1;
        state <= 3'b010;
      end
      // Calculation is done
      3'b100  : begin
        done <= 1;
        state <= 3'b100;
        if(~MPPT_search_done)
          state <= 3'b000;
      end
      default : begin
        state <= 3'b000;
        done <= 0;
      end
    endcase
  end

endmodule
```

D.3 I2C control

```
module I2CControl(
  input clk,
  input reset,
  input read_ref,
  input [10:0] result,
  inout sda,
  inout scl,
  output reg [7:0] ADC_out,
  output reg [1:0] CP_bits,
  output reg [3:0] CM_bits,
  output reg done,
  output reg [2:0] FB,
  output reg [10:0] reference,
  output ack_error
);


  wire busy;
  wire [7:0] data_rd;
  reg ena, rw, reset_n, start, read_ref_buf;      // rw: 0 = write
  reg [2:0] read_msg;
  reg [3:0] state;
  reg [4:0] count_read, count_write;
  reg [6:0] addr;
  reg [7:0] data_wr;
  reg [21:0] count_delay;
  reg [39:0] databuf;

  // I2C Master, controls the I2C lines (SDA/SCL)
  i2c_master I2Cm(
    .clk(clk),
    .reset_n(reset_n),
    .ena(ena),
    .addr(addr),
    .rw(rw),
    .data_wr(data_wr),
    .busy(busy),
    .data_rd(data_rd),
    .ack_error(ack_error),
    .sda(sda),
    .scl(scl)
  );


  // Sets ADC_out, CP_bits, and CM_bits from read data in databuf
  always @(posedge clk) begin
    if(reset) begin
        ADC_out <= 0;
        CP_bits <= 0;
        CM_bits <= 0;
        reference <= 0;
    end
    else begin
        ADC_out <= databuf[7:0];
```

```
      CP_bits <= databuf[9:8];
      CM_bits <= databuf[19:16];
      reference <= {databuf[36:32], databuf[29:24]};
  end
end


// Checks for the I2C start condition, in the same way as the I2C Master does
always @(negedge sda)
  start <= scl;


// Main FSM
always @(posedge clk) begin
  if(reset) begin
    state <= 4'b0000;
    ena <= 0;
    reset_n <= 0;
    done <= 0;
    read_msg <= 0;
    read_ref_buf <= 0;
    count_delay <= 0;
    count_read <=  0;
    count_write <= 0;
    addr <= 60;
    FB <= 0;
    databuf <= 0;
  end
  else begin
    case(state)
      // Initialize
      4'b0000 :  begin
          FB <= 1;
          read_msg <= 0;
          done <= 0;
          ena <= 1;
          state <= 4'b0001;
          reset_n <= 1;
          read_ref_buf <= read_ref;
          addr = 60;
          count_delay <= 0;
          count_read <= 0;
          count_write <= 0;
      end
      // Setup for write
      4'b0001 :  begin
         FB <= 2;
        ena <= 1;
        done <= 0;
        rw <= 0;
        reset_n <= 1;
        addr <= 60;
        data_wr <= {3'b000, result[10:6]};

        if(start)
          state <= 4'b0010;
```

```
  else
    state <= 4'b0001;
end
// Wait for write to finish
4'b0010 :  begin
    FB <= 3;
    done <= 0;
    rw <= 0;
    reset_n <= 1;
    addr = 60;
    ena <= 1;
    if (count_write > 0)          // Transition to stop command
        ena <= 0;
    if (count_write == 0)
        data_wr <= {2'b00, result[5:0]};

    // Wait for I2C Master to not be busy anymore, meaning it has send its message,
    ↪  then continue
    if (~busy)
        state <= 4'b0011;
    else
        state <= 4'b0010;
end

// Get ready for next message, or continue to read phase
4'b0011 : begin
    FB <= 4;
    done <= 0;
    rw <= 0;
    reset_n <= 1;
    if (count_write < 1) begin
        if (busy)
            state <= 4'b0100;
        else
            state <= 4'b0011;
    end
    else
        state <= 4'b0101;
end
// Increase count_write
4'b0100 :   begin
    FB <= 5;
    done <= 0;
    rw <= 0;
    reset_n <= 1;
    state <= 4'b0010;
    count_write <= count_write + 1;
end
// Delay between send and receive
4'b0101 :    begin   // Wait state
    FB <= 6;
    done <= 0;
    rw <= 1;
    ena <= 0;
    reset_n <= 1;
    count_write <= 0;
```

```
    if (count_delay == 4194303)
        state <= 4'b0110;
    else
        count_delay <= count_delay + 1;
end

// Setup for read
4'b0110 :  begin
  FB <= 7;
  done <= 0;
  rw <= 1;
  ena <= 1;
  reset_n <= 1;
  addr = 60;
  count_delay <= 0;
  count_read <= 0;
  if (start)
    state <= 4'b0111;
  else
    state <= 4'b0110;
end
// Wait for read to finish
4'b0111 :  begin
    done <= 0;
    rw <= 1;
    reset_n <= 1;
    addr = 60;
    if (count_read > 3)
        ena <= 0;
    else
        ena <= 1;

    if(~busy)
        state <= 4'b1000;
    else
        state <= 4'b0111;
end
4'b1000 :  begin
  done <= 0;
  rw <= 1;
  reset_n <= 1;
  addr = 60;

  if(count_read == 0)
    databuf[7:0] <= data_rd;
  else if (count_read == 1)
    databuf[15:8] <= data_rd;
  else if (count_read == 2)
    databuf[23:16] <= data_rd;
  else if (count_read == 3)
    databuf[31:24] <= data_rd;
  else if (count_read == 4)
    databuf[39:32] <= data_rd;

    // Transition to stop
    if (count_read > 3)
```

```
      ena <= 0;
    else
      ena <= 1;

    // If all messages are read
    if (count_read < 4) begin
        if (busy)
            state <= 4'b1001;
        else
            state <= 4'b1000;
    end
    else
      state <= 4'b1010;

end
// Increase count_read
4'b1001 :   begin
    done <= 0;
    rw <= 0;
    reset_n <= 1;
    addr <= 60;
    state <= 4'b0111;
    count_read <= count_read + 1;
end
4'b1010 :    begin
  done <= 1;
  rw <= 1;
  ena <= 0;
  reset_n <= 1;
  addr = 60;
  state <= 4'b1011;

end
4'b1011 :   begin
    done <= 1;
    rw <= 1;
    ena <= 0;
    reset_n <= 1;
    addr = 60;
    state <= 4'b1011;
    if (count_delay == 4194303)      //1048575
        state <= 4'b0000;
    else
        count_delay <= count_delay + 1;
end
default : begin
    done <= 0;
    rw <= 0;
    reset_n <= 0;
    ena <= 0;
    addr = 60;
    state <= 4'b0000;
    count_delay <= 0;
    count_read <= 0;
    count_write <= 0;
end
```

```
      endcase
    end
  end

endmodule
```

D.4 I2C master

```
--------------------------------------------------------------------------------
--
--   FileName:         i2c_master.vhd
--   Dependencies:     none
--   Design Software:  Quartus II 64-bit Version 13.1 Build 162 SJ Full Version
--
--   HDL CODE IS PROVIDED "AS IS."  DIGI-KEY EXPRESSLY DISCLAIMS ANY
--   WARRANTY OF ANY KIND, WHETHER EXPRESS OR IMPLIED, INCLUDING BUT NOT
--   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
--   PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL DIGI-KEY
--   BE LIABLE FOR ANY INCIDENTAL, SPECIAL, INDIRECT OR CONSEQUENTIAL
--   DAMAGES, LOST PROFITS OR LOST DATA, HARM TO YOUR EQUIPMENT, COST OF
--   PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY OR SERVICES, ANY CLAIMS
--   BY THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF),
--   ANY CLAIMS FOR INDEMNITY OR CONTRIBUTION, OR OTHER SIMILAR COSTS.
--
--   Version History
--   Version 1.0 11/01/2012 Scott Larson
--     Initial Public Release
--   Version 2.0 06/20/2014 Scott Larson
--     Added ability to interface with different slaves in the same transaction
--     Corrected ack_error bug where ack_error went 'Z' instead of '1' on error
--     Corrected timing of when ack_error signal clears
--   Version 2.1 10/21/2014 Scott Larson
--     Replaced gated clock with clock enable
--     Adjusted timing of SCL during start and stop conditions
--   Version 2.2 02/05/2015 Scott Larson
--     Corrected small SDA glitch introduced in version 2.1
--
--------------------------------------------------------------------------------

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;

ENTITY i2c_master IS
  GENERIC(
    input_clk : INTEGER := 5_000_000; --input clock speed from user logic in Hz
    bus_clk   : INTEGER := 100_000);   --speed the i2c bus (scl) will run at in Hz
    ↪  //400_000
  PORT(
    clk       : IN     STD_LOGIC;                    --system clock
    reset_n   : IN     STD_LOGIC;                    --active low reset
    ena       : IN     STD_LOGIC;                    --latch in command
    addr      : IN     STD_LOGIC_VECTOR(6 DOWNTO 0); --address of target slave
    rw        : IN     STD_LOGIC;                    --'0' is write, '1' is read
    data_wr   : IN     STD_LOGIC_VECTOR(7 DOWNTO 0); --data to write to slave
    busy      : OUT    STD_LOGIC;                    --indicates transaction in progress
    data_rd   : OUT    STD_LOGIC_VECTOR(7 DOWNTO 0); --data read from slave
    ack_error : BUFFER STD_LOGIC;                    --flag if improper acknowledge from slave
    sda       : INOUT  STD_LOGIC;                    --serial data output of i2c bus
    scl       : INOUT  STD_LOGIC);                   --serial clock output of i2c bus
END i2c_master;
```

```
ARCHITECTURE logic OF i2c_master IS
  CONSTANT divider  :  INTEGER := (input_clk/bus_clk)/4; --number of clocks in 1/4 cycle of
  ↪ scl
  TYPE machine IS(ready, start, command, slv_ack1, wr, rd, slv_ack2, mstr_ack, stop); --needed
  ↪ states
  SIGNAL state         : machine;                        --state machine
  SIGNAL data_clk      : STD_LOGIC;                      --data clock for sda
  SIGNAL data_clk_prev : STD_LOGIC;                      --data clock during previous system
  ↪ clock
  SIGNAL scl_clk       : STD_LOGIC;                      --constantly running internal scl
  SIGNAL scl_ena       : STD_LOGIC := '0';               --enables internal scl to output
  SIGNAL sda_int       : STD_LOGIC := '1';               --internal sda
  SIGNAL sda_ena_n     : STD_LOGIC;                      --enables internal sda to output
  SIGNAL addr_rw       : STD_LOGIC_VECTOR(7 DOWNTO 0);   --latched in address and read/write
  SIGNAL data_tx       : STD_LOGIC_VECTOR(7 DOWNTO 0);   --latched in data to write to slave
  SIGNAL data_rx       : STD_LOGIC_VECTOR(7 DOWNTO 0);   --data received from slave
  SIGNAL bit_cnt       : INTEGER RANGE 0 TO 7 := 7;      --tracks bit number in transaction
  SIGNAL stretch       : STD_LOGIC := '0';               --identifies if slave is stretching
  ↪ scl
BEGIN

  --generate the timing for the bus clock (scl_clk) and the data clock (data_clk)
  PROCESS(clk, reset_n)
    VARIABLE count  :  INTEGER RANGE 0 TO divider*4;  --timing for clock generation
  BEGIN
    IF(reset_n = '0') THEN                --reset asserted
      stretch <= '0';
      count := 0;
    ELSIF(clk'EVENT AND clk = '1') THEN
      data_clk_prev <= data_clk;          --store previous value of data clock
      IF(count = divider*4-1) THEN        --end of timing cycle
        count := 0;                       --reset timer
      ELSIF(stretch = '0') THEN           --clock stretching from slave not detected
        count := count + 1;               --continue clock generation timing
      END IF;
      CASE count IS
        WHEN 0 TO divider-1 =>            --first 1/4 cycle of clocking
          scl_clk <= '0';
          data_clk <= '0';
        WHEN divider TO divider*2-1 =>    --second 1/4 cycle of clocking
          scl_clk <= '0';
          data_clk <= '1';
        WHEN divider*2 TO divider*3-1 =>  --third 1/4 cycle of clocking
          scl_clk <= '1';                 --release scl
          IF(scl = '0') THEN              --detect if slave is stretching clock
            stretch <= '1';
          ELSE
            stretch <= '0';
          END IF;
          data_clk <= '1';
        WHEN OTHERS =>                    --last 1/4 cycle of clocking
          scl_clk <= '1';
          data_clk <= '0';
      END CASE;
    END IF;
  END PROCESS;
```

```
--state machine and writing to sda during scl low (data_clk rising edge)
PROCESS(clk, reset_n)
BEGIN
  IF(reset_n = '0') THEN                --reset asserted
    state <= ready;                     --return to initial state
    busy <= '1';                        --indicate not available
    scl_ena <= '0';                     --sets scl high impedance
    sda_int <= '1';                     --sets sda high impedance
    ack_error <= '0';                   --clear acknowledge error flag
    bit_cnt <= 7;                       --restarts data bit counter
    data_rd <= "00000000";              --clear data read port
  ELSIF(clk'EVENT AND clk = '1') THEN
    IF(data_clk = '1' AND data_clk_prev = '0') THEN  --data clock rising edge
      CASE state IS
        WHEN ready =>                       --idle state
          IF(ena = '1') THEN                --transaction requested
            busy <= '1';                    --flag busy
            addr_rw <= addr & rw;           --collect requested slave address and command
            data_tx <= data_wr;             --collect requested data to write
            state <= start;                 --go to start bit
          ELSE                              --remain idle
            busy <= '0';                    --unflag busy
            state <= ready;                 --remain idle
          END IF;
        WHEN start =>                       --start bit of transaction
          busy <= '1';                      --resume busy if continuous mode
          sda_int <= addr_rw(bit_cnt);      --set first address bit to bus
          state <= command;                 --go to command
        WHEN command =>                     --address and command byte of transaction
          IF(bit_cnt = 0) THEN              --command transmit finished
            sda_int <= '1';                 --release sda for slave acknowledge
            bit_cnt <= 7;                   --reset bit counter for "byte" states
            state <= slv_ack1;              --go to slave acknowledge (command)
          ELSE                              --next clock cycle of command state
            bit_cnt <= bit_cnt - 1;         --keep track of transaction bits
            sda_int <= addr_rw(bit_cnt-1); --write address/command bit to bus
            state <= command;               --continue with command
          END IF;
        WHEN slv_ack1 =>                    --slave acknowledge bit (command)
          IF(addr_rw(0) = '0') THEN         --write command
            sda_int <= data_tx(bit_cnt);    --write first bit of data
            state <= wr;                    --go to write byte
          ELSE                              --read command
            sda_int <= '1';                 --release sda from incoming data
            state <= rd;                    --go to read byte
          END IF;
        WHEN wr =>                          --write byte of transaction
          busy <= '1';                      --resume busy if continuous mode
          IF(bit_cnt = 0) THEN              --write byte transmit finished
            sda_int <= '1';                 --release sda for slave acknowledge
            bit_cnt <= 7;                   --reset bit counter for "byte" states
            state <= slv_ack2;              --go to slave acknowledge (write)
          ELSE                              --next clock cycle of write state
            bit_cnt <= bit_cnt - 1;         --keep track of transaction bits
            sda_int <= data_tx(bit_cnt-1); --write next bit to bus
```

```
          state <= wr;                    --continue writing
        END IF;
      WHEN rd =>                          --read byte of transaction
        busy <= '1';                      --resume busy if continuous mode
        IF(bit_cnt = 0) THEN              --read byte receive finished
          IF(ena = '1' AND addr_rw = addr & rw) THEN  --continuing with another read at
          ↪  same address
            sda_int <= '0';               --acknowledge the byte has been received
          ELSE                            --stopping or continuing with a write
            sda_int <= '1';               --send a no-acknowledge (before stop or repeated
            ↪  start)
          END IF;
          bit_cnt <= 7;                   --reset bit counter for "byte" states
          data_rd <= data_rx;             --output received data
          state <= mstr_ack;              --go to master acknowledge
        ELSE                              --next clock cycle of read state
          bit_cnt <= bit_cnt - 1;         --keep track of transaction bits
          state <= rd;                    --continue reading
        END IF;
      WHEN slv_ack2 =>                    --slave acknowledge bit (write)
        IF(ena = '1') THEN                --continue transaction
          busy <= '0';                    --continue is accepted
          addr_rw <= addr & rw;           --collect requested slave address and command
          data_tx <= data_wr;             --collect requested data to write
          IF(addr_rw = addr & rw) THEN    --continue transaction with another write
            sda_int <= data_wr(bit_cnt); --write first bit of data
            state <= wr;                  --go to write byte
          ELSE                            --continue transaction with a read or new slave
            state <= start;               --go to repeated start
          END IF;
        ELSE                              --complete transaction
          state <= stop;                  --go to stop bit
        END IF;
      WHEN mstr_ack =>                    --master acknowledge bit after a read
        IF(ena = '1') THEN                --continue transaction
          busy <= '0';                    --continue is accepted and data received is
          ↪  available on bus
          addr_rw <= addr & rw;           --collect requested slave address and command
          data_tx <= data_wr;             --collect requested data to write
          IF(addr_rw = addr & rw) THEN    --continue transaction with another read
            sda_int <= '1';               --release sda from incoming data
            state <= rd;                  --go to read byte
          ELSE                            --continue transaction with a write or new slave
            state <= start;               --repeated start
          END IF;
        ELSE                              --complete transaction
          state <= stop;                  --go to stop bit
        END IF;
      WHEN stop =>                        --stop bit of transaction
        busy <= '0';                      --unflag busy
        state <= ready;                   --go to idle state
    END CASE;
  ELSIF(data_clk = '0' AND data_clk_prev = '1') THEN  --data clock falling edge
    CASE state IS
      WHEN start =>
        IF(scl_ena = '0') THEN                  --starting new transaction
```

```
            scl_ena <= '1';                     --enable scl output
            ack_error <= '0';                   --reset acknowledge error output
          END IF;
        WHEN slv_ack1 =>                        --receiving slave acknowledge (command)
          IF(sda /= '0' OR ack_error = '1') THEN  --no-acknowledge or previous
          ↪  no-acknowledge
            ack_error <= '1';                   --set error output if no-acknowledge
          END IF;
        WHEN rd =>                              --receiving slave data
          data_rx(bit_cnt) <= sda;              --receive current slave data bit
        WHEN slv_ack2 =>                        --receiving slave acknowledge (write)
          IF(sda /= '0' OR ack_error = '1') THEN  --no-acknowledge or previous
          ↪  no-acknowledge
            ack_error <= '1';                   --set error output if no-acknowledge
          END IF;
        WHEN stop =>
          scl_ena <= '0';                       --disable scl
        WHEN OTHERS =>
          NULL;
      END CASE;
    END IF;
  END IF;
END PROCESS;

--set sda output
WITH state SELECT
  sda_ena_n <= data_clk_prev WHEN start,     --generate start condition
               NOT data_clk_prev WHEN stop,  --generate stop condition
               sda_int WHEN OTHERS;          --set to internal sda signal

--set scl and sda outputs
scl <= '0' WHEN (scl_ena = '1' AND scl_clk = '0') ELSE 'Z';
sda <= '0' WHEN sda_ena_n = '0' ELSE 'Z';

END logic;
```

E.
Excel output calculation results

Lux	ADC output	Number out	Simulated lux	Relative error	Absolute error
9543	129	846	9533	0.00105	10
9617	130	847	9533	0.00873	84
9691	131	848	9637	0.00557	54
9765	132	848	9742	0.00236	23
9839	133	849	9848	0.00091	9
9913	134	850	9848	0.00656	65
9987	135	850	9955	0.00320	32
10061	136	851	10060	0.00010	1
10135	137	852	10060	0.00740	75
10209	138	852	10170	0.00382	39
10283	139	853	10280	0.00029	3
10357	140	854	10280	0.00743	77
10431	141	854	10400	0.00297	31
10505	142	855	10510	0.00048	5
10579	143	856	10510	0.00652	69
10653	144	856	10620	0.00310	33
10727	145	857	10740	0.00121	13
10801	146	858	10740	0.00565	61
10875	147	858	10860	0.00138	15
10949	148	859	10860	0.00813	89
11023	149	859	10970	0.00481	53
11097	150	860	11090	0.00063	7
11170	151	861	11090	0.00716	80
11244	152	861	11210	0.00302	34
11318	153	862	11210	0.00954	108
11392	154	862	11340	0.00456	52
11466	155	863	11460	0.00052	6
11540	156	864	11460	0.00693	80
11614	157	864	11590	0.00207	24
11688	158	865	11590	0.00838	98
11762	159	865	11710	0.00442	52
11836	160	866	11840	0.00034	4
11910	161	867	11840	0.00588	70
11984	162	867	11970	0.00117	14
12058	163	868	11970	0.00730	88
12132	164	868	12100	0.00264	32
12206	165	869	12100	0.00868	106

12280	166	869	12230	0.00407	50
12354	167	870	12360	0.00049	6
12428	168	870	12360	0.00547	68
12502	169	871	12500	0.00016	2
12576	170	872	12500	0.00604	76
12650	171	872	12630	0.00158	20
12724	172	873	12630	0.00739	94
12798	173	873	12770	0.00219	28
12872	174	874	12770	0.00792	102
12946	175	874	12910	0.00278	36
13020	176	875	12910	0.00845	110
13094	177	875	13050	0.00336	44
13168	178	876	13050	0.00896	118
13242	179	876	13190	0.00393	52
13316	180	877	13190	0.00946	126
13390	181	877	13340	0.00373	50
13464	182	878	13340	0.00921	124
13538	183	878	13480	0.00428	58
13612	184	879	13630	0.00132	18
13686	185	879	13630	0.00409	56
13760	186	880	13780	0.00145	20
13834	187	880	13780	0.00390	54
13908	188	881	13780	0.00920	128
13982	189	881	13930	0.00372	52
14056	190	882	13930	0.00896	126
14130	191	882	14080	0.00354	50
14204	192	883	14080	0.00873	124
14278	193	883	14230	0.00336	48
14351	194	884	14230	0.00843	121
14425	195	884	14390	0.00243	35
14499	196	885	14390	0.00752	109
14573	197	885	14540	0.00226	33
14647	198	886	14540	0.00731	107
14721	199	886	14700	0.00143	21
14795	200	887	14700	0.00642	95
14869	201	887	14860	0.00061	9
14943	202	888	14860	0.00555	83
15017	203	888	15020	0.00020	3
15091	204	888	15020	0.00470	71
15165	205	889	15020	0.00956	145
15239	206	889	15190	0.00322	49
15313	207	890	15190	0.00803	123
15387	208	890	15350	0.00240	37
15461	209	891	15350	0.00718	111
15535	210	891	15520	0.00097	15
15609	211	892	15520	0.00570	89
15683	212	892	15690	0.00045	7
15757	213	892	15690	0.00425	67
15831	214	893	15690	0.00891	141

15905	215	893	15860	0.00283	45
15979	216	894	15860	0.00745	119
16053	217	894	16030	0.00143	23
16127	218	895	16030	0.00601	97
16201	219	895	16210	0.00056	9
16275	220	895	16210	0.00399	65
16349	221	896	16210	0.00850	139
16423	222	896	16380	0.00262	43
16497	223	897	16380	0.00709	117
16571	224	897	16560	0.00066	11
16645	225	897	16560	0.00511	85
16719	226	898	16560	0.00951	159
16793	227	898	16740	0.00316	53
16867	228	899	16740	0.00753	127
16941	229	899	16930	0.00065	11
17015	230	899	16930	0.00500	85
17089	231	900	17110	0.00123	21
17163	232	900	17110	0.00309	53
17237	233	901	17110	0.00737	127
17311	234	901	17300	0.00064	11
17385	235	901	17300	0.00489	85
17459	236	902	17300	0.00911	159
17532	237	902	17480	0.00297	52
17606	238	903	17480	0.00716	126
17680	239	903	17670	0.00057	10
17754	240	903	17670	0.00473	84
17828	241	904	17670	0.00886	158
17902	242	904	17870	0.00179	32
17976	243	905	17870	0.00590	106
18050	244	905	18060	0.00055	10
18124	245	905	18060	0.00353	64
18198	246	906	18060	0.00758	138
18272	247	906	18260	0.00066	12
18346	248	906	18260	0.00469	86
18420	249	907	18260	0.00869	160
18494	250	907	18460	0.00184	34
18568	251	908	18460	0.00582	108
18642	252	908	18660	0.00097	18
18716	253	908	18660	0.00299	56
18790	254	909	18660	0.00692	130
18864	255	909	18660	0.01081	204

F.

Excel output calculation sheet

Reference	Ref log	Reff	Reff log	CP ratio	CP ratio log	ADC log	Temp out	C param	Out	Out full	ADC output	Lux	Simulation	Error	Error
9543	13.22	731.96	9.5156	2	1	7.0112	6.495606	14.7246	13.22	846	129	9543	9533	0.10%	10
		731.96	9.5156	2	1	7.0112		14.7246	13.22	846	129	9543	9533	0.10%	10
		731.96	9.5156	2	1	7.0224		14.7246	13.231	847	130	9617	9533	0.87%	84
		731.96	9.5156	2	1	7.0334		14.7246	13.242	848	131	9691	9637	0.56%	54
		731.96	9.5156	2	1	7.0444		14.7246	13.253	848	132	9765	9742	0.24%	23
		731.96	9.5156	2	1	7.0553		14.7246	13.264	849	133	9839	9848	0.09%	9
		731.96	9.5156	2	1	7.0661		14.7246	13.275	850	134	9913	9848	0.66%	65
		731.96	9.5156	2	1	7.0768		14.7246	13.286	850	135	9987	9955	0.32%	32
		731.96	9.5156	2	1	7.0875		14.7246	13.296	851	136	10061	10060	0.01%	1
		731.96	9.5156	2	1	7.098		14.7246	13.307	852	137	10135	10060	0.74%	75
		731.96	9.5156	2	1	7.1085		14.7246	13.318	852	138	10209	10170	0.38%	39
		731.96	9.5156	2	1	7.1189		14.7246	13.328	853	139	10283	10280	0.03%	3
		731.96	9.5156	2	1	7.1293		14.7246	13.338	854	140	10357	10280	0.74%	77
		731.96	9.5156	2	1	7.1396		14.7246	13.349	854	141	10431	10400	0.30%	31
		731.96	9.5156	2	1	7.1497		14.7246	13.359	855	142	10505	10510	0.05%	5
		731.96	9.5156	2	1	7.1599		14.7246	13.369	856	143	10579	10510	0.65%	69
		731.96	9.5156	2	1	7.1699		14.7246	13.379	856	144	10653	10620	0.31%	33
		731.96	9.5156	2	1	7.1799		14.7246	13.389	857	145	10727	10740	0.12%	13
		731.96	9.5156	2	1	7.1898		14.7246	13.399	858	146	10801	10740	0.56%	61
		731.96	9.5156	2	1	7.1997		14.7246	13.409	858	147	10875	10860	0.14%	15
		731.96	9.5156	2	1	7.2095		14.7246	13.418	859	148	10949	10860	0.81%	89
		731.96	9.5156	2	1	7.2192		14.7246	13.428	859	149	11023	10970	0.48%	53
		731.96	9.5156	2	1	7.2288		14.7246	13.438	860	150	11097	11090	0.06%	7
		731.96	9.5156	2	1	7.2384		14.7246	13.447	861	151	11170	11090	0.72%	80
		731.96	9.5156	2	1	7.2479		14.7246	13.457	861	152	11244	11210	0.30%	34
		731.96	9.5156	2	1	7.2574		14.7246	13.466	862	153	11318	11210	0.95%	108
		731.96	9.5156	2	1	7.2668		14.7246	13.476	862	154	11392	11340	0.46%	52
		731.96	9.5156	2	1	7.2761		14.7246	13.485	863	155	11466	11460	0.05%	6
		731.96	9.5156	2	1	7.2854		14.7246	13.494	864	156	11540	11460	0.69%	80
		731.96	9.5156	2	1	7.2946		14.7246	13.504	864	157	11614	11590	0.21%	24
		731.96	9.5156	2	1	7.3038		14.7246	13.513	865	158	11688	11590	0.84%	98
		731.96	9.5156	2	1	7.3129		14.7246	13.522	865	159	11762	11710	0.44%	52

		731.96	9.5156	2	1	7.3219		14.7246	13.531	866	160	11836	11840	0.03%	4
		731.96	9.5156	2	1	7.3309		14.7246	13.54	867	161	11910	11840	0.59%	70
		731.96	9.5156	2	1	7.3399		14.7246	13.549	867	162	11984	11970	0.12%	14
		731.96	9.5156	2	1	7.3487		14.7246	13.558	868	163	12058	11970	0.73%	88
		731.96	9.5156	2	1	7.3576		14.7246	13.567	868	164	12132	12100	0.26%	32
		731.96	9.5156	2	1	7.3663		14.7246	13.575	869	165	12206	12100	0.87%	106
		731.96	9.5156	2	1	7.375		14.7246	13.584	869	166	12280	12230	0.41%	50
		731.96	9.5156	2	1	7.3837		14.7246	13.593	870	167	12354	12360	0.05%	6
		731.96	9.5156	2	1	7.3923		14.7246	13.601	870	168	12428	12360	0.55%	68
		731.96	9.5156	2	1	7.4009		14.7246	13.61	871	169	12502	12500	0.02%	2
		731.96	9.5156	2	1	7.4094		14.7246	13.618	872	170	12576	12500	0.60%	76
		731.96	9.5156	2	1	7.4179		14.7246	13.627	872	171	12650	12630	0.16%	20
		731.96	9.5156	2	1	7.4263		14.7246	13.635	873	172	12724	12630	0.74%	94
		731.96	9.5156	2	1	7.4346		14.7246	13.644	873	173	12798	12770	0.22%	28
		731.96	9.5156	2	1	7.4429		14.7246	13.652	874	174	12872	12770	0.79%	102
		731.96	9.5156	2	1	7.4512		14.7246	13.66	874	175	12946	12910	0.28%	36
		731.96	9.5156	2	1	7.4594		14.7246	13.668	875	176	13020	12910	0.84%	110
		731.96	9.5156	2	1	7.4676		14.7246	13.677	875	177	13094	13050	0.34%	44
		731.96	9.5156	2	1	7.4757		14.7246	13.685	876	178	13168	13050	0.90%	118
		731.96	9.5156	2	1	7.4838		14.7246	13.693	876	179	13242	13190	0.39%	52
		731.96	9.5156	2	1	7.4919		14.7246	13.701	877	180	13316	13190	0.95%	126
		731.96	9.5156	2	1	7.4998		14.7246	13.709	877	181	13390	13340	0.37%	50
		731.96	9.5156	2	1	7.5078		14.7246	13.717	878	182	13464	13340	0.92%	124
		731.96	9.5156	2	1	7.5157		14.7246	13.725	878	183	13538	13480	0.43%	58
		731.96	9.5156	2	1	7.5236		14.7246	13.733	879	184	13612	13630	0.13%	18
		731.96	9.5156	2	1	7.5314		14.7246	13.74	879	185	13686	13630	0.41%	56
		731.96	9.5156	2	1	7.5392		14.7246	13.748	880	186	13760	13780	0.15%	20
		731.96	9.5156	2	1	7.5469		14.7246	13.756	880	187	13834	13780	0.39%	54
		731.96	9.5156	2	1	7.5546		14.7246	13.764	881	188	13908	13780	0.92%	128
		731.96	9.5156	2	1	7.5622		14.7246	13.771	881	189	13982	13930	0.37%	52
		731.96	9.5156	2	1	7.5699		14.7246	13.779	882	190	14056	13930	0.90%	126
		731.96	9.5156	2	1	7.5774		14.7246	13.786	882	191	14130	14080	0.35%	50
		731.96	9.5156	2	1	7.585		14.7246	13.794	883	192	14204	14080	0.87%	124

		731.96	9.5156	2	1	7.5925		14.7246	13.801	883	193	14278	14230	0.34%	48
		731.96	9.5156	2	1	7.5999		14.7246	13.809	884	194	14351	14230	0.84%	121
		731.96	9.5156	2	1	7.6073		14.7246	13.816	884	195	14425	14390	0.24%	35
		731.96	9.5156	2	1	7.6147		14.7246	13.824	885	196	14499	14390	0.75%	109
		731.96	9.5156	2	1	7.6221		14.7246	13.831	885	197	14573	14540	0.23%	33
		731.96	9.5156	2	1	7.6294		14.7246	13.838	886	198	14647	14540	0.73%	107
		731.96	9.5156	2	1	7.6366		14.7246	13.846	886	199	14721	14700	0.14%	21
		731.96	9.5156	2	1	7.6439		14.7246	13.853	887	200	14795	14700	0.64%	95
		731.96	9.5156	2	1	7.6511		14.7246	13.86	887	201	14869	14860	0.06%	9
		731.96	9.5156	2	1	7.6582		14.7246	13.867	888	202	14943	14860	0.56%	83
		731.96	9.5156	2	1	7.6653		14.7246	13.874	888	203	15017	15020	0.02%	3
		731.96	9.5156	2	1	7.6724		14.7246	13.881	888	204	15091	15020	0.47%	71
		731.96	9.5156	2	1	7.6795		14.7246	13.888	889	205	15165	15020	0.96%	145
		731.96	9.5156	2	1	7.6865		14.7246	13.896	889	206	15239	15190	0.32%	49
		731.96	9.5156	2	1	7.6935		14.7246	13.902	890	207	15313	15190	0.80%	123
		731.96	9.5156	2	1	7.7004		14.7246	13.909	890	208	15387	15350	0.24%	37
		731.96	9.5156	2	1	7.7074		14.7246	13.916	891	209	15461	15350	0.72%	111
		731.96	9.5156	2	1	7.7142		14.7246	13.923	891	210	15535	15520	0.10%	15
		731.96	9.5156	2	1	7.7211		14.7246	13.93	892	211	15609	15520	0.57%	89
		731.96	9.5156	2	1	7.7279		14.7246	13.937	892	212	15683	15690	0.04%	7
		731.96	9.5156	2	1	7.7347		14.7246	13.944	892	213	15757	15690	0.43%	67
		731.96	9.5156	2	1	7.7415		14.7246	13.95	893	214	15831	15690	0.89%	141
		731.96	9.5156	2	1	7.7482		14.7246	13.957	893	215	15905	15860	0.28%	45
		731.96	9.5156	2	1	7.7549		14.7246	13.964	894	216	15979	15860	0.74%	119
		731.96	9.5156	2	1	7.7616		14.7246	13.971	894	217	16053	16030	0.14%	23
		731.96	9.5156	2	1	7.7682		14.7246	13.977	895	218	16127	16030	0.60%	97
		731.96	9.5156	2	1	7.7748		14.7246	13.984	895	219	16201	16210	0.06%	9
		731.96	9.5156	2	1	7.7814		14.7246	13.99	895	220	16275	16210	0.40%	65
		731.96	9.5156	2	1	7.7879		14.7246	13.997	896	221	16349	16210	0.85%	139
		731.96	9.5156	2	1	7.7944		14.7246	14.003	896	222	16423	16380	0.26%	43
		731.96	9.5156	2	1	7.8009		14.7246	14.01	897	223	16497	16380	0.71%	117
		731.96	9.5156	2	1	7.8074		14.7246	14.016	897	224	16571	16560	0.07%	11
		731.96	9.5156	2	1	7.8138		14.7246	14.023	897	225	16645	16560	0.51%	85

		731.96	9.5156	2	1	7.8202		14.7246	14.029	898	226	16719	16560	0.95%	159
		731.96	9.5156	2	1	7.8265		14.7246	14.036	898	227	16793	16740	0.32%	53
		731.96	9.5156	2	1	7.8329		14.7246	14.042	899	228	16867	16740	0.75%	127
		731.96	9.5156	2	1	7.8392		14.7246	14.048	899	229	16941	16930	0.06%	11
		731.96	9.5156	2	1	7.8455		14.7246	14.054	899	230	17015	16930	0.50%	85
		731.96	9.5156	2	1	7.8517		14.7246	14.061	900	231	17089	17110	0.12%	21
		731.96	9.5156	2	1	7.858		14.7246	14.067	900	232	17163	17110	0.31%	53
		731.96	9.5156	2	1	7.8642		14.7246	14.073	901	233	17237	17110	0.74%	127
		731.96	9.5156	2	1	7.8704		14.7246	14.079	901	234	17311	17300	0.06%	11
		731.96	9.5156	2	1	7.8765		14.7246	14.086	901	235	17385	17300	0.49%	85
		731.96	9.5156	2	1	7.8826		14.7246	14.092	902	236	17459	17300	0.91%	159
		731.96	9.5156	2	1	7.8887		14.7246	14.098	902	237	17532	17480	0.30%	52
		731.96	9.5156	2	1	7.8948		14.7246	14.104	903	238	17606	17480	0.72%	126
		731.96	9.5156	2	1	7.9009		14.7246	14.11	903	239	17680	17670	0.06%	10
		731.96	9.5156	2	1	7.9069		14.7246	14.116	903	240	17754	17670	0.47%	84
		731.96	9.5156	2	1	7.9129		14.7246	14.122	904	241	17828	17670	0.89%	158
		731.96	9.5156	2	1	7.9189		14.7246	14.128	904	242	17902	17870	0.18%	32
		731.96	9.5156	2	1	7.9248		14.7246	14.134	905	243	17976	17870	0.59%	106
		731.96	9.5156	2	1	7.9307		14.7246	14.14	905	244	18050	18060	0.06%	10
		731.96	9.5156	2	1	7.9366		14.7246	14.146	905	245	18124	18060	0.35%	64
		731.96	9.5156	2	1	7.9425		14.7246	14.152	906	246	18198	18060	0.76%	138
		731.96	9.5156	2	1	7.9484		14.7246	14.157	906	247	18272	18260	0.07%	12
		731.96	9.5156	2	1	7.9542		14.7246	14.163	906	248	18346	18260	0.47%	86
		731.96	9.5156	2	1	7.96		14.7246	14.169	907	249	18420	18260	0.87%	160
		731.96	9.5156	2	1	7.9658		14.7246	14.175	907	250	18494	18460	0.18%	34
		731.96	9.5156	2	1	7.9715		14.7246	14.181	908	251	18568	18460	0.58%	108
		731.96	9.5156	2	1	7.9773		14.7246	14.186	908	252	18642	18660	0.10%	18
		731.96	9.5156	2	1	7.983		14.7246	14.192	908	253	18716	18660	0.30%	56
		731.96	9.5156	2	1	7.9887		14.7246	14.198	909	254	18790	18660	0.69%	130

G.
Used MATLAB code

G.1 ModelInit.m

```
% clear all;
clear vvec;

global k;
global q;
global T0;
global G0;
global Vt;

% Constants
k = 1.38e-23;
q = 1.602177e-19;

% Design parameters
T0 = 298.15;
G0 = 1000;
Vt = k*T0/q;

SCtype = 6;         % Solar cell type: 0 = Panasonic, 1 = IXOLAR, 3 = Panasonic basic, 4 =
↪  AM-1456, 6 = custom AM-1456
Tcor = 0;           % Have G calculation correct for different temperature or not
inout = 1;          % 0 = use output voltage, 1 = use input voltage
actual = 1;         % Use actual input current
fourfive = 0;       % 0 = 4 parameter model, 1 = 5 parameter model

%G = 0.4396*1000;  % Irradiance in suns (1000 W/m^2)        (for panasonic curve 2)
G = 1000;
T = 25 + (T0-25);    % Actual temperature

M = 10000;          % Current mirror ratio
Rsense = 1;         % Sense resistor value
W = 1e-2;           % Transistor width (matters for accuracy)
L = 1e-4;           % Transistor length

% Tolerances and inefficiencies
Rout = 1;           % Charge pump output resistance
dRout = 1;          % Charge pump output resistance error factor
dRsense = 1;        % Sense resistor error factor
dN = 1;             % Transformer ratio error factor
dM = 1;             % SenseFET ratio error factor
```

```
Voff = 0;           % SenseFET amplifier offset voltage
CMRatio = 1;        % Current mirror ratio

% Datasheet parameters
% Panasonic Amorton
% 0.4396 for panasonic curve 2
if SCtype == 0
    N = 1.28;           % Voltage gain
    % N = 1.15;         % -20
    % N = 1.29;           % lower curve (G = 0.4396)
    Voc = 5.34;%5.34;
    Isc = 5.46e-3;
    Vmpp = 3.9;
    Impp = 4.6e-3;
    Pmpp = 18e-3;
    alpha = 0.0000044; % Isc temperature coefficient    A/deg C
    beta = -0.0163; % Voc temperature coefficient       V/deg C
    ns = 6;
end

% IXOLAR
if SCtype == 1
    % N = 11.3;           % Voltage gain --> blijkbaar niet
    N = 8.95;               % Voltage gain
    % N = 10;
    Voc = .69;
    Isc = 58.6e-3;
    Vmpp = .56;
    Impp = 55e-3;
    Pmpp = 30.8e-3;
    alpha = 26.5e-6; % Isc temperature coefficient
    beta = -0.0017; % Voc temperature coefficient
    ns = 1;
end

% PVmodel paper
if SCtype == 2
    Isc = 3.87;
    Impp = 3.56;
    Vmpp = 33.7;
    Voc = 42.1;
    alpha = 3.13;
    beta = -0.16;
    ns = 72;
    % Rs = 0.47;
    % Rsh = 1365;
    % A = 1.397;
end

% Panasonic Amorton basic
if SCtype == 3
    Voc = .89;
    Isc = 14.8e-3;
    Vmpp = .64;
    Impp = 12.4e-3;
    Pmpp = 7.89e-3;
```

```
    alpha = 26.5e-6; % Isc temperature coefficient
    beta = -0.0017; % Voc temperature coefficient
    ns = 1;
end


% Panasonic Amorton AM-1456 (indoor)
% Curves op G = 0.0018*G0 en 4.396e-4*G0
if SCtype == 4
%     G0 = 1000*0.0018;
%     G = G0;
    N = 1.28;           % Voltage gain
    Voc = 2.4;
    Isc = 6e-6*3;
    Vmpp = 1.5;
    Impp = 5.3e-6*3;
    Pmpp = 7.95e-6*3;
    alpha = 0.005e-6; % Isc temperature coefficient    A/deg C
    beta = -0.0112; % Voc temperature coefficient
    ns = 4;
end

if SCtype == 5
    G0 = 200;                   % Lux
    G = 1000;
    N = 1.28;                   % Voltage gain
%     Voc = 2.5;                  % Catalog
%     Isc = 35.2e-6;
    % Vmpp = 1.5;               % Catalog
    % Impp = 33.3e-6;
    Voc = 2.4;                 % Datasheet
    Isc = 35e-6;
    Vmpp = 1.5;                % Datasheet
    Impp = 31e-6;
%     Vmpp = 1.91;              % Datasheet curve
%     Impp = 2.747e-5;

    Pmpp = Vmpp*Impp;

    alpha = 1.75e-7; % Isc temperature coefficient    A/deg C
    beta = -4.18e-4; % Voc temperature coefficient
    ns = 4;
end


if SCtype == 6
%     G0 = 1000*0.0018;
%     G = G0;
factor = 20;
    Voc = 2.4+Vt*1.5*4*log(factor);
    Isc = 18e-6*factor;
    Vmpp = 1.5+Vt*1.5*4*log(factor)*0.85;
    Impp = 5.3e-6*factor*0.85*3;
    Pmpp = Impp*Vmpp;
    alpha = 0.005e-6; % Isc temperature coefficient    A/deg C
    beta = -0.0112; % Voc temperature coefficient
```

```
    ns = 4;
end


% Calculated parameters
% A = 1.3408;
% Rs = 0.6171;
% A = 3.42;
% Rs = 3.84;
Rsh = 99999999999;

% if ((exist('A','var') ~= 1) || (exist('Rs','var') ~= 1))
if (fourfive == 0)
    [A, Rs] = fourparammodel(Voc, Isc, Vmpp, Impp, ns)
    disp('A and Rs updated')
else
    [A, Rs, Rsh] = fiveparammodel(Voc, Isc, Vmpp, Impp, Pmpp, ns)
    disp('A, Rs, and Rsh updated')
end

clear Gout;
clear varvec;

    [Voc0, Isc0, Io0, Iph0, VocG, IscG, Io, Iph, ivec, vvec]  = PV_curves(Voc, Isc, A, Rs,
    ↪  Rsh, alpha, beta, ns, G, T);
    disp('Curve updated')


Tsend = Tcor*T + (~Tcor * T0);

[a, b] = MPPfind(vvec, ivec)
```

G.2 PV_curves.m

```
function [Voc0, Isc0, Io0, Iph0, Voc, Isc, Io, Iph, ivec, vvec] = PV_curves(Voc, Isc, A, Rs,
↪  Rsh, alpha, beta, ns, G, T)

global k;
global q;
global T0;
global G0;
global Vt;

% Expressions
Io = (Isc - (Voc-Isc*Rs)/Rsh)*exp(-Voc/Vt/A/ns);
Iph = Io*exp(Voc/Vt/A/ns) + Voc/Rsh;

% Calculate temperature effects
VocT = Voc + beta*(T-T0);
IscT = Isc + alpha * (T-T0);
IoT = (IscT - (VocT - IscT*Rs)/Rsh)*exp(-VocT/Vt/A/ns);
IphT = IoT * exp(VocT/Vt/A/ns) + VocT/Rsh;
% Calculate irradiance effects
IscG = IscT*G/G0;
IphG = IphT*G/G0;


%syms Vocx;
VocG = VocT+Vt*A*ns*log(G/G0);


Voc0 = Voc;
Isc0 = Isc;
Io0 = Io;
Iph0 = Iph;

Voc = VocG;
Isc = IscG;
Io = IoT;
Iph = IphG;

% % I-V curve
% syms ix
% vvec = linspace(0,6,300);
% for x = 1:length(vvec)
%     v = vvec(x);
%     eq4 = Iph - Io*(exp((v+ix*Rs)/(Vt*A*ns)) - 1) - (v + ix*Rs)/Rsh;
%     ivec(x) = double(vpasolve(eq4 == ix));
% end
% %
% plot(vvec, ivec);
% grid on;
% % ylim([0 6e-3])
%
% set(gca, 'YLim', [0, get(gca, 'YLim') * [0; 1]])


% V-I curve instead of I-V
```

```
syms vx
ivec = linspace(0,Isc,300);


for x = 1:length(ivec)
     i = ivec(x);
%     vvec(x) = Vt*A*ns*log(i/Iph + (1 - i/Iph)*exp(Voc/Vt/A/ns)) - Rs*i;

    eq4 = Iph - Io*(exp((vx+i*Rs)/(Vt*A*ns)) - 1) - (vx + i*Rs)/Rsh;

    vvec(x) = double(vpasolve(eq4 == i));

end
plot(vvec, ivec);
grid on;
hold on;
end
```

G.3 fourparammodel.m

```
% Four parameter model from model18DirectEquationForVtRs
function [A, Rs] = fourparammodel(Voc, Isc, Vmpp, Impp, ns)

% Constants
k = 1.38e-23;
q = 1.602177e-19;
T0 = 298.15;
G0 = 1000;

% Environment
T = 25+(T0-25);
G = G0;

% Direct parameter calculation
Vt = ((2*Vmpp-Voc)*(Isc-Impp)) / (Impp + (Isc-Impp)*log(1-Impp/Isc)); % = Vt*A*ns
Rs = Vmpp/Impp - (2*Vmpp-Voc)/(Impp+(Isc-Impp)*log(1-Impp/Isc));
A = Vt/ns/k/T0*q;
```

G.4 fiveparammodel.m

```
function [A, Rs, Rsh] = fiveparammodel(Voc, Isc, Vmpp, Impp, Pmpp, ns)

global k;
global q;
global T0;
global G0;
global Vt;

% Environment
T = 25+(T0-25);
G = G0;

A = 1;
tol = 0.00001;
step = 0.0001;
iter = 100000;
it = 0;

Vt = k*T/q;     % 6
Rs = Voc/Impp + Vt*A*ns/Impp * log(Vt*A*ns/(Vt*A*ns+Vmpp))-Vmpp/Impp;
I0 = Isc / (exp(Voc/Vt/A/ns) - exp(Rs*Isc/Vt/A/ns));
Iph = I0 * (exp(Voc/Vt/A/ns) - 1);
VmppC = Vt*A*ns * log((Iph + I0 - Impp)/I0) - Rs*Impp;
err = abs(VmppC - Vmpp);

oscdet = 0;

while(err > tol && it < iter)
    if VmppC < Vmpp
        A = A-step;
        if oscdet < 0
            oscdet = oscdet*-1 + 1;
        else
            oscdet = 1;
        end
    else
        A = A+step;
        if oscdet > 0
            oscdet = oscdet*-1 -1;
        else
            oscdet = -1;
        end
    end
    I0 = Isc / (exp(Voc/Vt/A/ns) - exp(Rs*Isc/Vt/A/ns));
    Iph = I0 * (exp(Voc/Vt/A/ns) - 1); % 9
    VmppC = Vt*A*ns * log((Iph + I0 - Impp)/I0) - Rs*Impp;

    err = abs(VmppC - Vmpp);
    it = it + 1;

    if abs(oscdet) > 10
        break
    end
```

```
end

Rs = Voc/Impp + Vt*A*ns/Impp * log(Vt*A*ns/(Vt*A*ns+Vmpp))-Vmpp/Impp;

tol = 0.00000001;
step = 10;
iter = 1000;
itI = 0;

Rsh = Vmpp*(Vmpp + Rs*Impp) / (Vmpp * Iph - Vmpp * I0 * (exp((Vmpp+Rs * Impp)/Vt/A/ns) - 1) -
↪  Pmpp);
I0 = Isc / (exp(Voc/Vt/A/ns) - exp(Rs*Isc/Vt/A/ns)); % 10
Iph = I0 * (exp(Voc/Vt/A/ns) - 1); % 9

syms ImppCT
eq5 = Iph - I0 * (exp((Vmpp+Rs*ImppCT)/Vt/A/ns) - 1) - (Vmpp+Rs*ImppCT)/Rsh;
ImppC = double(vpasolve(eq5 == ImppCT));

oscdet = 0;

% for h = 1:iter
while(err > tol && itI < iter)
    if ImppC < Impp
        Rsh = Rsh+step;
        if oscdet < 0
            oscdet = oscdet*-1 + 1;
        else
            oscdet = 1;
        end
    else
        Rsh = Rsh-step;
        if oscdet > 0
            oscdet = oscdet*-1 -1;
        else
            oscdet = -1;
        end
    end

    I0 = (Isc * (1+ Rs/Rsh) - Voc/Rsh) / (exp(Voc/Vt/A/ns) - exp(Rs*Isc/Vt/A/ns));
    Iph = I0 * (exp(Voc/Vt/A/ns) - 1) + Voc/Rsh;

    eq5 = Iph - I0 * (exp((Vmpp+Rs*ImppCT)/Vt/A/ns) - 1) - (Vmpp+Rs*ImppCT)/Rsh;
    ImppC = double(vpasolve(eq5 == ImppCT));

    err = abs(ImppC - Impp);
    itI = itI+1;

    if (abs(oscdet) > 10)
        break;
    end
end

disp(A);
disp(Rs);
disp(Rsh);
```

H.
Cadence Genus synthesis file

```
set_attribute lib_search_path
↪  /home/project/Atlas/DigitalFlow/0105633/TSMCHOME/digital/Front_End/timing_power_noise/
↪  NLDM/tcb018bcdgp2a_110a

set_attribute library {tcb018bcdgp2awc.lib}

read_hdl /home/olmar/Desktop/Verilogarithmisch/square_comb.v
read_hdl /home/olmar/Desktop/Verilogarithmisch/dec_log_conv_red.v
read_hdl /home/olmar/Desktop/Verilogarithmisch/log_LUT.v
read_hdl /home/olmar/Desktop/Verilogarithmisch/log_ill_calc_FSM.v
read_hdl /home/olmar/Desktop/Verilogarithmisch/log_top_level.v

elaborate
create_clock -period 1000 -name clock2 [get_ports clk]

synthesize -to_mapped

write -mapped >
↪  /home/olmar/Desktop/Verilogarithmisch/synthesis/log_top_level_synth_genus_1000.v

gui_show
```